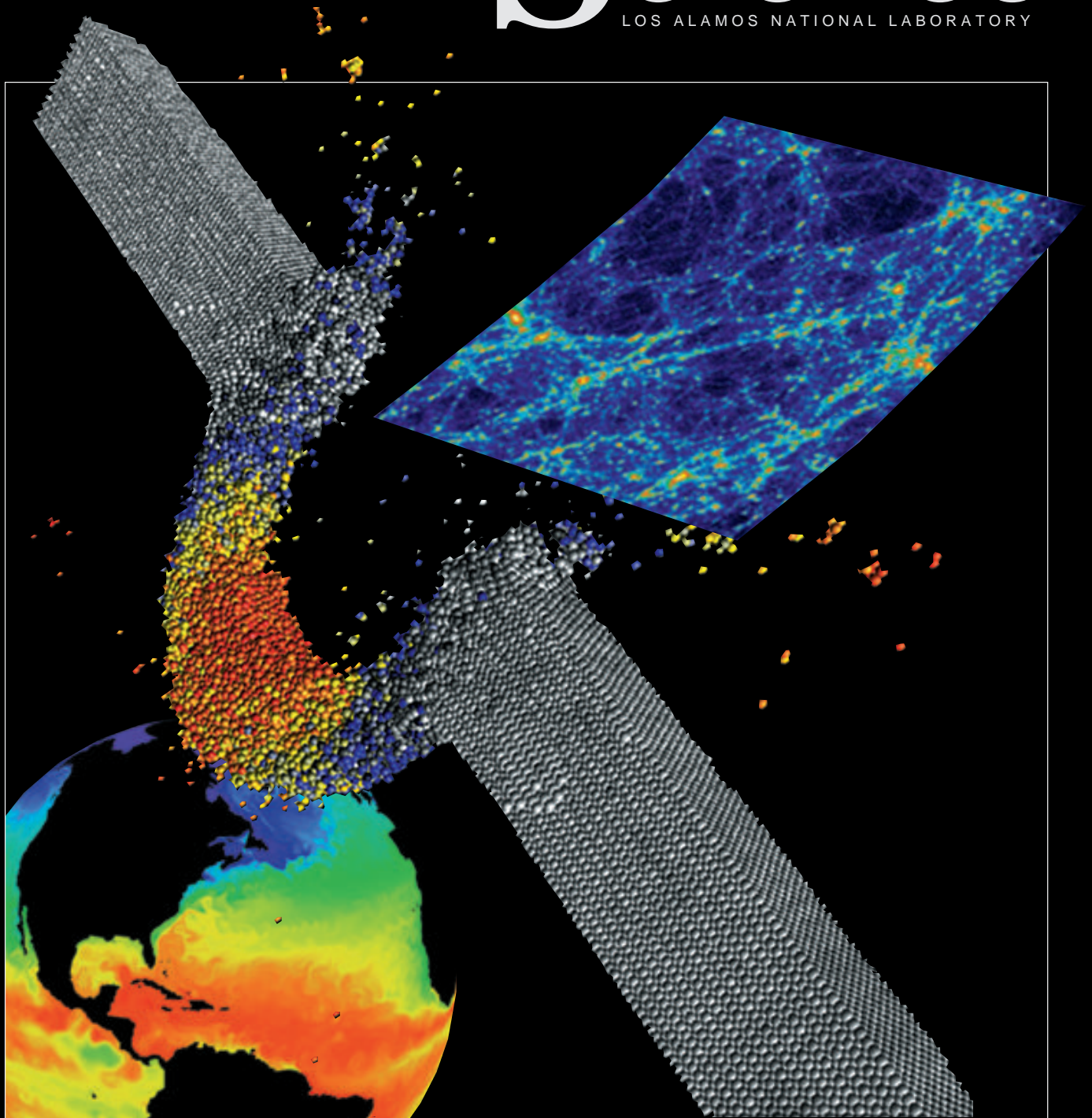


Los Alamos Science

LOS ALAMOS NATIONAL LABORATORY



Number 22 1994

High-Performance Computing



Laboratory scientists have made enormous progress in performing large-scale numerical simulations on massively parallel computers. The images on the cover are frames taken from three such simulations. At the upper right is an image from a simulation of the formation of large-scale structure in an expanding universe. The simulation involves millions of particles representing both luminous and cold dark matter and is being used to compare observations with the predictions of cosmological models. The center image is from a molecular-dynamics simulation and shows a solid undergoing fracture. Multi-million-atom simulations such as this will be used to study the physics of materials. At the lower left is a frame from the simulation of global circulation patterns in the ocean. Simulations of ocean circulation are being combined with those describing atmospheric motions to model the long-term dynamics of climate.

Editor

Necia Grant Cooper

Managing Editor

Nadine Shea

Science Writers

Gerald A. Friedman

Douglas D. Lemon

Sheila K. Schiferl

Nancy K. Shera

Art Director

Gloria Sharp

Technical Illustration

Andrea J. Kron

Cartoons

Donald R. Montoya

Photography

John A. Flower

Enrique F. Ortega

Computer Art

James M. Cruz

Anita L. Flores

Eric A. Vigil

Production

Nadine Shea

Other Contributors

Kay P. Coddens

AnnMarie Dyson

Susanne M. Kornke

Cheri Tiedman-Isham

CIC-9 Photography

Printing

Guadalupe D. Archuleta

Address Mail to:

Los Alamos Science

Mail Stop M708

Los Alamos National Laboratory

Los Alamos, NM 87545

lascience@lanl.gov

Windows on Computing: New Initiatives at Los Alamos 1
David W. Forslund, Charles A. Slocumb, and Ira A. Agins

History of Computers at Los Alamos 2

Collaborations with Industry on Parallel Computing 12
Bruce R. Wienke

How Computers Work: An Introduction to Serial, Vector, and Parallel Computers 14
Gerald A. Friedman, Douglas D. Lemon, and Tony T. Warnock

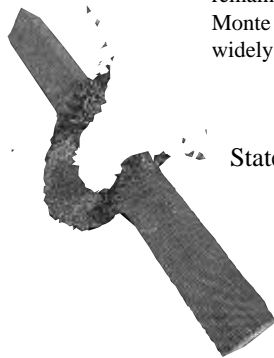


All electronic computers are composed of simple elements that perform simple operations. This article explains the way those elements work together and describes the differences between serial, vector, and parallel supercomputers.

HIPPI – The First Standard for High-Performance Networking 26
Stephen C. Tenbrink and Donald E. Tolmie

A Monte Carlo Code for Particle Transport – An Algorithm for All Seasons 30
John S. Hendricks

The Monte Carlo method, invented at the Laboratory in the 1940s, remains one of the most versatile numerical techniques. MCNP, a Monte Carlo particle transport code is one of the Laboratory's most widely used products.



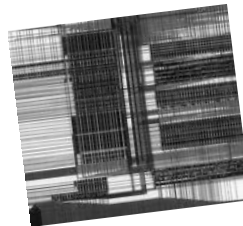
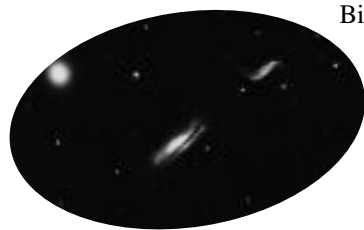
State-of-the-Art Parallel Computing – Molecular Dynamics on the Connection Machine 44
Peter S. Lomdahl and David M. Beazley

Realizing the performance capabilities of the massively parallel CM-5 supercomputer for real problems is a major challenge to computational scientists. This article describes how molecular-dynamics methods for materials science were optimized for the CM-5.

Experimental Cosmology and the Puzzle of Large-Scale Structure 58
Wojciech H. Zurek and Michael S. Warren

Big Bang Cosmology and the Microwave Background 82
Salman Habib and Raymond J. Laflamme

A Fast Tree Code for Many-Body Problems 88
Michael S. Warren and John K. Salmon



The Laboratory has been in the forefront of large-scale scientific computing since before the invention of electronic computers. It is now a leader in the shift to parallel computing, in the development of collaborative relationships with industry, and in the development of data-management tools for use on the nation's future information highways.

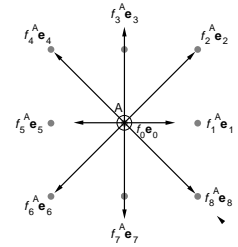


Stars make up galaxies, which make up galaxy clusters, which make up still larger structures. Theories of how these structures originated are being tested against observations through the use of "experimental cosmology"—numerical simulations on massively parallel computers that accurately follow the motions of tens of millions of massive particles under various sets of assumptions.

Lattice-Boltzmann Fluid Dynamics – A Versatile Tool for Multiphase and Other Complicated Flows 98

Shiyi Chen, Gary D. Doolen, and Kenneth G. Eggert

The lattice-Boltzmann method uses a simple set of kinetic rules to describe the motion of particles on a lattice. It yields informative and computationally efficient simulations of fluid flow, particularly for complex processes such as the flow of oil and water through porous rock—a process of great interest to the oil industry.



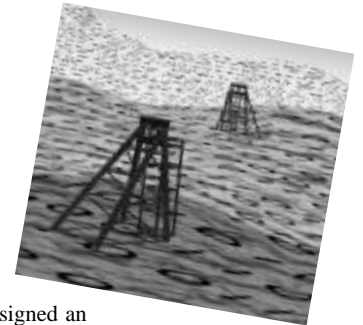
Equations of the Lattice-Boltzmann Method 110



Toward Improved Prediction of Reservoir Flow Performance —
Simulating Oil and Water Flows at the Pore Scale 112

John J. Buckles, Randy D. Hazlett, Shiyi Chen, Kenneth G. Eggert, Daryl W. Grunau, and Wendy E. Soll

Researchers from Mobil Corporation and the Laboratory are collaborating on lattice-Boltzmann simulations to predict basic parameters that determine reservoir flow performance.



Concept Extraction – A Data Mining Technique 122

Vance Faber, Judith G. Hochberg, Patrick M. Kelly, Timothy R. Thomas, and James M. White

Clustering and the Continuous *k*-Means Algorithm 138

Vance Faber

Extracting meaningful information from large datasets is a formidable task. The work can be efficiently divided between humans and computers, with each assigned an appropriate portion. Analysis is facilitated by using a powerful clustering algorithm along with a well-designed user interface.



The Digital Village Initiative 150

John D. MacCuish, Susan M. Mniszewski, Gregory E. Shannon, and Bonnie C. Yantis

In response to the National Information Infrastructure initiative, the Laboratory is collaborating with developers of local telecommunications—computer networks that provide services and facilitate communications.

@xxx.lanl.gov — First Steps Toward Electronic Research Communication 156

Paul H. Ginsparg

A Laboratory scientist originated and implemented the idea of making a continually updated database of preprints accessible to users around the world. Such databases have become a very popular medium for scientific communication.



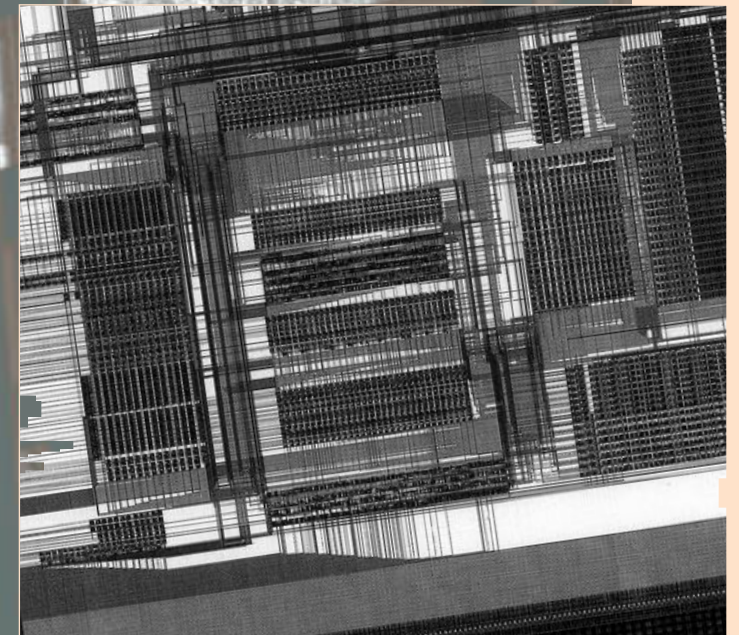


Windows on Computing

New Initiatives at Los Alamos

David W. Forslund, Charles A. Slocomb, and Ira A. Agins

No aspect of technology is changing more rapidly than the field of computing and information systems. It is among the fastest growing and most competitive arenas in our global economy. Each year, more of the items we use contain tiny microprocessors—silicon chips on which are etched hundreds or thousands or millions of electronic circuit elements. Those computer chips direct various operations and adjustments—automatic braking in cars; automatic focusing in cameras; automatic data collection in cash registers; automatic message-taking by answering machines; automatic operation of washers, dryers, and other appliances; automatic production of goods in manufacturing plants; the list could go on and on. Those inconspicuous devices that perform micro-scale computing are profoundly shaping our lives and our culture.



Opening illustration: Elements of high-performance computing: Left, the CM-5 Connection Machine, the most powerful massively parallel supercomputer at Los Alamos; center, the foyer in the Laboratory Data Communications Center (LDCC); upper right, digitized images of Washington, D.C. from a wavelet-based multiresolution database developed at Los Alamos; and lower right, a portion of a “metacomputer,” a 950,000-transistor special-purpose chip for analyzing the behavior of digital circuits. The chip is being developed by a team of graduate students at the University of Michigan.

More visible and no less important are the ways microprocessors are changing the way we communicate with each other and even the kinds of tasks we do. Industries such as desktop publishing, electronic mail, multimedia systems, and financial accounting systems have been created by the ubiquitous microprocessor. It is nothing short of the engine of the digital revolution.

The computer chip was invented in 1958 when Jack Kilby figured out how to fabricate several transistors on a single-crystal silicon substrate and thereby created the integrated circuit. Since then more and more transistors have been integrated on a single chip. By the early 1980s the technology of VLSI, or very-large scale integration (hundreds of thousands of transistors on

a chip), had led to dramatic reductions in the cost of producing powerful microprocessors and large memory units. As a result affordable personal computers and powerful workstations have become commonplace in science, in business, and in the home.

New microprocessors continue to be incorporated into various products at an increasing rate; development cycles are down to months rather than years as the current generation of processors are used to aid in the design and manufacture of the next generation. Because of their economies of scale, off-the-shelf microprocessors are expanding the use of micro- and medium-scale computing in business and in the home. They are also motivating changes in the design

History of Computers at Los Alamos

1943–45

Desktop calculators and punched-card accounting machines are used as calculating tools in the Manhattan Project.

1945

ENIAC, the world's first large-scale electronic computer, is completed at the University of Pennsylvania. Its “shake-down” calculation is the “Los Alamos problem,” a calculation needed for the design of thermonuclear weapons.

1949

IBM's first Card Programmable Calculators are installed at the Laboratory.

1952

MANIAC is built at the Laboratory under the direction of Nick Metropolis. It is the first computer designed from the start according to John von Neumann's stored-program ideas.

1953

The Laboratory gets serial number 2 of the IBM 701. This “Defense Calculator” is approximately equal in power to the MANIAC.

1955

The MANIAC II project, a computer featuring floating-point arithmetic, is started. The Laboratory begins working closely with computer manufacturers to ensure that its future computing needs will be satisfied.



MANIAC II

1956

MANIAC II is completed. The Laboratory installs serial number 1 of the the IBM 704, which has about the same power as MANIAC II. From this point on, the Laboratory acquires supercomputers from industry.

Late 1950s

The Laboratory and IBM enter into a joint project to build STRETCH, a computer based on transistors rather than vacuum tubes, to meet the needs of the nuclear-weapons program.

1961

STRETCH is completed and is about thirty-five times as powerful as the IBM 704. IBM used much of the technology developed for STRETCH in its computers for years afterward.

1966

The first on-line mass-storage system with a capacity of over 10^{12} bits, the IBM 1360 Photo Store, is installed at the Laboratory. Control Data Corporation introduces the first “pipelined” computer, the CDC 6600, designed by Seymour Cray. The Laboratory buys a few.

1971

The Laboratory buys its first CDC 7600, the successor to the 6600. These machines are the main supercomputers in use at the Laboratory during much of the 1970s.

1972

Cray Research, Inc is founded. The Laboratory consults on the design of the Cray-1.

1975

Laboratory scientists design and build a high-speed network that uses 50-megabit-per-second channels.

of large-scale scientific computers. The focus has shifted from single, very fast processors to very fast networks that allow hundreds to thousands of microprocessors to cooperate on a single problem. Large-scale computation is a critical technology in scientific research, in major industries, and in the maintenance of national security. It is also the area of computing in which Los Alamos has played a major role.

The microprocessor has opened up the possibility of continual increases in the power of supercomputers through the architecture of the MPP, the massively parallel processor that can consist of thousands of off-the-shelf microprocessors. In 1989, seeing the potential of that new technology for ad-

ressing the “Grand Challenge” computational problems in science and engineering, Los Alamos set up the Advanced Computing Laboratory as a kind of proving ground for testing MPPs on real problems. Ironically, just as their enormous potential is being clearly demonstrated at Los Alamos and elsewhere, economic forces stemming from reduced federal budgets and slow acceptance into the commercial marketplace are threatening the viability of the supercomputing industry.

As a leader in scientific computing, Los Alamos National Laboratory has always understood the importance of supercomputing for maintaining national security and economic strength. At this critical juncture the Laboratory

plans to continue working with the supercomputing industry and to help expand the contributions of computer modeling and simulation to all areas of society. Here, we will briefly review a few of our past contributions to the high end of computing, outline some new initiatives in large-scale parallel computing, and then introduce a relatively new area of involvement, our support of the information revolution and the National Information Infrastructure initiative.

Since the days of the Manhattan Project, Los Alamos has been a driver of and a major participant in the development of large-scale scientific computation. It was here that Nick Metropolis directed the construction of MANIAC I

1976

Serial number 1 of the Cray-1 is delivered to the Laboratory.

1977

A Common File System, composed of IBM mass-storage components, is installed and provides storage for all central and remote Laboratory computer systems.



The Cray-1

1980

The Laboratory begins its parallel-processing efforts.

1981

An early parallel processor (PuPS) is fabricated at the Laboratory but never completed.

1983

Denelcor's HEP, an early commercially available parallel processor, is installed, as is the first of five Cray X-MP computers.

1985

The Ultra-High-Speed Graphics Project is started. It pioneers animation as a visualization tool and requires gigabit-per-second communication capacity. A massively parallel (128-node) Intel computer is installed.

1987

The need for higher communication capacity is answered by the development of the High-Performance Parallel Interface (HIPPI), an 800-megabit/second channel, which becomes an ANSI standard.

1988

The Laboratory obtains the first of its six Cray Y-MP computers. It also installs, studies, and evaluates a number of massively parallel computers. The Advanced Computing Laboratory (ACL) is established.

1989

The ACL purchases the CM-2 Connection Machine from Thinking Machines. It has 65,536 parallel processors.

1990

A test device for HIPPI ports is transferred to industry. The Laboratory, the Jet Propulsion Laboratory, and the San Diego Supercomputer start the Casa Gigabit Test Project

1991

The Laboratory transfers to industry the HIPPI framebuffer, an important component for visualization of complex images.

1992

A 1024-processor Thinking Machines CM-5, the most powerful computer at the time, is installed at the ACL.

1994

A massively parallel Cray T3D is installed at the ACL for use in collaborations with industry.



The Cray T3D

and II. Maniac I (1952) was among the first general-purpose digital computers to realize von Neumann's concept of a stored-program computer—one that could go from step to step in a computation by using a set of instructions that was stored electronically in its own memory in the same way that data are stored. (In contrast, the ENIAC (1945), the very first General-purpose electronic computer, had to be programmed mechanically by inserting cables into a plugboard.) Maniac II (1956) embodied another major advancement in computer design, the ability to perform floating-point arithmetic—the kind that automatically keeps track of the position of the decimal point. The peak speed of MANIAC II was 10,000 arithmetic operations per second, an impressive factor of 1000 higher than that of the electromechanical accounting machines used to perform numerical calculations at Los Alamos during the Manhattan Project (see Figure 1).

The main function of those accounting machines and very early electronic computers was to simulate through numerical computation the extreme physical conditions and complex nonlinear processes that occur within a nuclear weapon. The continued role of computer simulation as the primary tool for designing and predicting the performance and safety of nuclear weapons has been a major driver behind the development of increasingly powerful computers. The larger the computer, the greater the complexity that could be simulated accurately. That is still true today as the largest supercomputers are being used to simulate realistically very complex phenomena including the interaction of the oceans and the atmosphere on global scales, the basis of bulk properties of materials in the motions of individual atoms, the flow of oil and gas through the porous rock of underground reservoirs, the dynamics of the

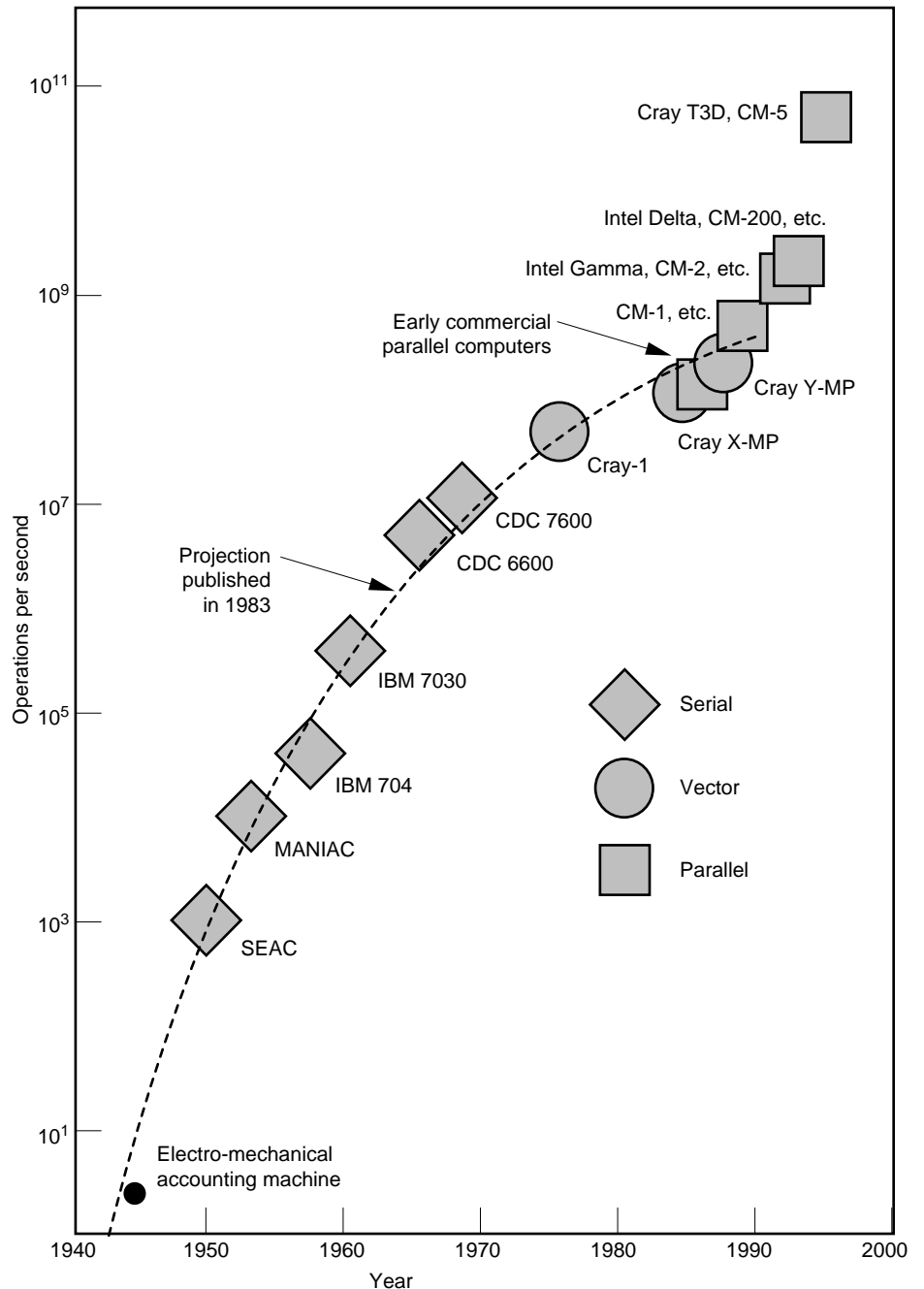


Figure 1. The Growth of Computing Power

This plot shows the number of operations per second versus year for the fastest available "supercomputers". Different shapes are used to distinguish serial, vector, and parallel architectures. All computers up until and including the Cray-1 were single-processor machines. The dashed line is a projection of supercomputing speed through 1990. That projection was published in a 1983 issue of *Los Alamos Science*—before massively parallel processors became practical.

internal combustion engine, the behavior of biological macromolecules as pharmaceuticals, and so forth.

After 1956 and MANIAC II the Laboratory stopped building computers and instead relied on close working relationships with IBM and other vendors to ensure that the industry would be able to supply the necessary computing power. A particularly important collaborative effort was STRETCH, a project with IBM started in 1956 to design and build the fastest computer possible with existing technology. For the first time the new semiconductor transistor would be used in computer design. Transistors have a much faster response time and are much more reliable than the traditional vacuum-tube elements of digital circuits. The STRETCH, or IBM 7030, computer was made from 150,000 transistors and was delivered to Los Alamos in 1961. It was about thirty-five times faster than the IBM 704, a commercial vacuum-tube machine similar to the MANIAC II.

In the early 1970s Los Alamos became a consultant to Seymour Cray on the design of the Cray-1, the first successful vector computer. Vector architecture increases computational speed by enabling the computer to perform many machine instructions at once on linear data arrays (vectors). In contrast, computers with traditional serial architecture perform one machine instruction at a time on individual pieces of data (see "How Computers Work" in this volume).

Los Alamos was not only a consultant on the design of the Cray but was also the first purchaser of that innovative hardware. The delivery of the first Cray computer to Los Alamos in 1976 might be said to mark the beginning of the modern era of high-performance computing. The Cray-1 supercomputer had a speed of tens of megaflops (one megaflop equals a million floating-point

operations per second) and a memory capacity of 4 megabytes.

In all these developments at the high end of computing, the Laboratory has taken part in the struggles associated with bringing new technology into the marketplace, testing new machines, developing the operating systems, and developing new computational techniques,

The Laboratory has taken part in the struggles associated with bringing new supercomputing technology into the marketplace, testing new machines, developing the operating systems, and developing new computational techniques, or algorithms, to make full use of the potential computing power presented by each new supercomputer. That type of involvement continues today.

Los Alamos was one of the first institutions to demonstrate the general usefulness of the CM-2 Connection Machine, a massively parallel computer built by Thinking Machines Corporation.

or algorithms, to make full use of the potential computing power presented by each new supercomputer. That type of involvement continues today. The Laboratory was one of the first institutions to demonstrate the general usefulness of

the CM-2 Connection Machine, a massively parallel computer built by Thinking Machines Corporation. The CM-2 was originally designed to investigate problems in artificial intelligence, and when it arrived in Los Alamos in 1988 to be tried out on problems related to atmospheric effects and other hydrodynamic problems of interest in the weapons program, it did not contain the special processors needed for efficient computation of floating-point arithmetic. Laboratory scientists responded by working in collaboration with Thinking Machines on the installation and testing of the appropriate processing units and went on to develop the first successful algorithms for performing hydrodynamic problems on the Connection Machine's parallel architecture.

The question of whether parallel architectures would be a practical, cost-effective path to ever-increasing computer power has been hanging in the balance since the early 1980s. By then it was apparent that the speed of a single vector processor was limited by the physical size and density of the elements in the integrated circuits and the time required for propagation of electronic signals across the machine.

To increase the speed or overall performance of supercomputers by more than the factor associated with the speed-up in the individual processors, one could use parallel systems, for example, a number of Cray-1-type vector processors working together or massively parallel systems in which thousands of less powerful processors communicate with each other. In the early 1980s the latter approach appeared to be intractable.

At the same time that supercomputer manufacturers were worrying about how to improve on the standards of performance set by the Cray-1, an effort was begun to apply VLSI technolo-

gy to the design of a “Cray on a chip.” That effort to produce very powerful microprocessors was motivated in the early 1980s by the rapid growth of and standardization in the PC and workstation marketplace. As microprocessor technology advanced and commercial applications increased, production costs of microprocessors decreased by orders of magnitude and it became possible to produce very high-performance workstations with price-to-performance ratios much lower than those associated with conventional vector supercomputers. The workstations produced by Sun, Silicon Graphics, IBM, and Hewlett-Packard can sustain speeds comparable to the Cray-1 (see Figure 2). Such performance is sufficient for a wide range of scientific problems, and many researchers are choosing to adapt their computing to high-end workstations and workstations networked together to form workstation clusters. The microprocessor revolution has led to the workstation revolution in scientific computing.

Supercomputer manufacturers have also tried to build on the cost savings and technology advances afforded by the microprocessor revolution. Thinking Machines, Intel, Cray Research, and others have been designing MPPs, each containing hundreds or thousands of off-the-shelf microprocessors of the kind found in high-end workstations. Those processors are connected in parallel through various network architectures and are meant to work simultaneously on different parts of a single large problem. As indicated in Figure 2, MPPs hold the promise of increasing performance by factors of thousands because many of their designs are scalable, that is, their computational speed increases in proportion to the number of processors in the machine. One of the largest scalable MPPs in use is at

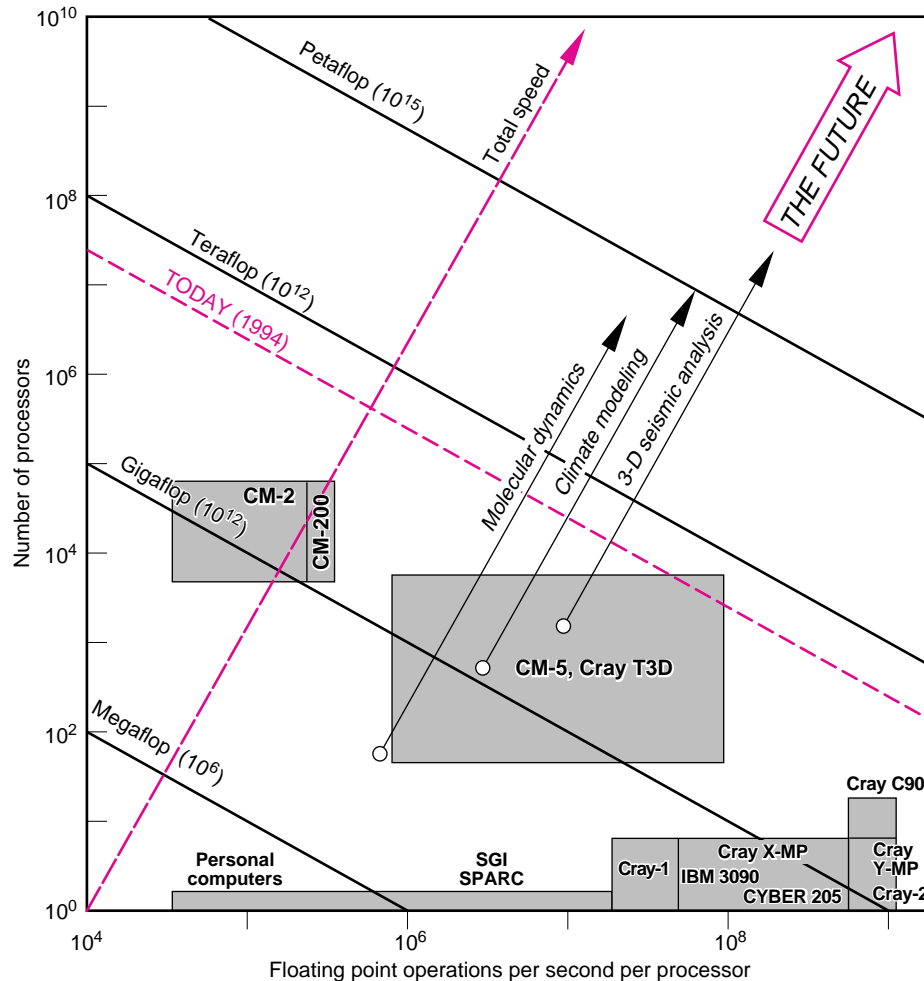


Figure 2. Supercomputers Today and Tomorrow

The speeds of workstations, vector supercomputers and massively parallel supercomputers are compared by plotting the number of processors versus the speed per processor measured in number of floating-point operations per second (flops). The solid diagonal lines are lines of constant total speed. The maximum theoretical speed of the 1024-node CM-5 Connection Machine is 131 gigaflops. Supercomputer manufacturers are hoping to make MPPs with teraflop speeds available within the next few years. Increases in speed will probably be achieved by increasing both the number of processors in an MPP and the speed per processor.

the Los Alamos Advanced Computing Laboratory—a CM-5 Connection Machine from Thinking Machines containing 1024 SPARC microprocessors. This machine has a theoretical peak speed of 130 gigaflops, more than a factor of 1000 over the Cray-1; some of

the applications (for example, the molecular dynamics calculation in “State-of-the-Art Parallel Computing”) run on the machine have already achieved 40 per cent of that theoretical maximum.

For over fifteen years the Laboratory has had much success with convention-

al vector supercomputers, particularly the Cray X-MP and Cray Y-MP. Los Alamos scientists have grown accustomed to “vectorizing” their computer programs, particularly hydrodynamics codes, and have been quite successful at designing codes that run very efficiently on the vector machines. Nevertheless, during the last five years the Laboratory has been strongly supporting the shift to parallel computing. It is our belief that the best hope of achieving the highest performance at the lowest cost lies with the massively parallel approach. In 1989 the Laboratory purchased its first Connection Machine, a CM-2, and in 1992 the Laboratory acquired the CM-5 mentioned above.

We have also been collaborating with manufacturers of high-end workstations on another approach to parallel computing, the development of workstation clusters. A cluster consists of many stand-alone workstations connected in a network that combines the computing power and memory capacity of the members of the cluster. We are helping to develop the networks and the software necessary to make such a cluster appear to the user as a single computational resource rather than a collection of individual workstations. Clusters are likely to come into greater and greater use because they provide scalable computing power at an excellent price-to-performance ratio starting from a small number of processors. At the higher performance end of computing, where hundreds of processors are involved, clusters do not necessarily compete well with vector or massively parallel supercomputers because the management of the processors becomes very complex and the scalability of the interconnections becomes more difficult to achieve.

Figure 3 shows two workstation clusters at Los Alamos. One, the IBM cluster consisting of eight IBM

RS/6000 560 workstations, is now in general use. The other, which is still in development, consists of eight Hewlett-Packard 735 workstations, each of which is faster at scalar calculations than is a single processor of a Cray Y-MP. We are now working closely with Hewlett-Packard on the software for that new cluster.

Achieving high performance on a massively parallel machine or a workstation cluster requires the use of entirely different programming models. The most flexible approach is the MIMD, or multiple-instruction, multiple-data, model in which different operations are performed simultaneously by different processors on different data. The challenge is to achieve communication and coordination among the processors without losing too much computational time. “State-of-the-Art Parallel Computing” presents a good look at what is involved in achieving high performance on MPPs.

The rich array of programming possibilities offered by parallel architectures has brought back into use many algorithms that had been ignored during the fifteen years or so that vector supercomputers dominated the scientific marketplace. Today the most active area in scientific programming and the one that will have a long-range impact on high-performance computing is the development of algorithms for parallel architectures. The Laboratory is a leader in algorithm development, and this volume presents a few outstanding examples of new parallel algorithms.

The Laboratory has also taken a leadership role in creating an advanced computing environment needed to achieve sustained high performance on the new MPPs and to store and view the enormous amounts of data generated by those machines. The focal point for research on and development and implementation of the elements of the



Figure 3. Workstation Clusters

Workstation clusters are becoming more popular for scientific computing because they have excellent price/performance ratios and can be upgraded as new microprocessors come on the market. The eight RS/6000 workstations in the IBM cluster at Los Alamos (top) are soon to be upgraded from model 560 to model 590. We are collaborating with Hewlett-Packard in the development of software for the very high-performance HP 735 workstation cluster recently assembled here (bottom). This cluster is being outfitted with HIPPI, the networking interface developed at the Laboratory for very high data transmission rates (see “HIPPI—The First Standard for High-Performance Networking”).

new computing environment is the Advanced Computing Laboratory set up at Los Alamos in 1989. The goal of the ACL is to handle the most challenging, computationally intensive problems in science and technology, the so-called Grand Challenge problems. Our 1024-processor CM-5, with its enormous speed and very large memory capacity, is the centerpiece of the Advanced Computing Laboratory. The ACL also houses advanced storage facilities developed at Los Alamos to rapidly store and retrieve the terabytes of data generated by running Grand Challenge problems on the CM-5. A special "HIPPI" network, developed at Los Alamos specifically to handle very high rates of data transmission, connects the CM-5 to the advanced storage facilities and vice versa. The HIPPI protocol for supercomputer networks has since become an industry standard (see "HIPPI").

Five DOE Grand Challenge problems are being investigated at the ACL: global climate modeling, multiphase flow, new materials technology, quantum chromodynamics, and the tokamak fusion reactor. Other very interesting simulations (see "Experimental Cosmology and the Puzzle of Large-scale Structures") are being performed on our CM-5 and have demonstrated the great potential of MPPs for scientific and engineering research.

As we make parallel computing work for Grand Challenge and other problems of fundamental interest, we are in a good position to help industry take advantage of the new computing opportunities. Our work with Mobil Corporation on modeling multiphase flow through porous media (see "Toward Improved Reservoir Flow Performance") uses an intrinsically parallel, Los Alamos-developed algorithm known as the lattice-Boltzmann method to model the flow of oil and water at

the scale of the individual pores in oil-bearing rock. The success of this collaboration provided a working example on which to base ACTI, the Advanced Computing Technology Initiative for

Today the most active area in scientific programming and the one that will have a long-range impact on high-performance computing is the development of algorithms for parallel architectures.

The Laboratory is a leader in algorithm development for the new parallel machines. Two outstanding examples described in this volume are the fast tree code for many-body problems, developed initially to trace the formation of structure in the early universe, and the lattice-Boltzmann method, an intrinsically parallel approach to the solution of complex multiphase flow problems of interest to the oil and gas industry.

establishing research collaborations between the oil and gas industry and the DOE national laboratories.

Another new collaborative effort, led by Los Alamos National Laboratory, Lawrence Livermore National Laboratory, and Cray Research, Inc., is specifically designed to transfer high-performance parallel-processing technology

to U.S. industry with the goal of increasing industrial competitiveness. The \$52 million collaborative program is under the auspices of the DOE's High Performance Parallel Processor program and will involve fifteen industrial partners. Over seventy scientists will be involved in creating computer algorithms for massively parallel machines that are of direct use in simulating complex industrial processes and their environmental impact.

In addition, two networked 128-processor Cray T3D MPPs, one at Los Alamos and one at Livermore, will be used in the industrial collaborations. They will be the first government-owned machines to be focused primarily on solving industrial problems.

The program includes fifteen projects in the fields of environmental modeling, petroleum exploration, materials design, advanced manufacturing, and new MPP systems software. Los Alamos will be involved in seven of the projects, two of which are devoted to developing general software tools and diagnostics that cut across particular applications and address important issues of portability of software from one computing platform to another. Those projects in which Los Alamos will participate are listed in the box "Collaborations with Industry on Parallel Computing."

So far we have concentrated on the Laboratory's initiatives in high-performance computing. But the digital revolution and the culture of the Internet has led us into a second major area—the National Information Infrastructure (NII) initiative. The goal of this federal initiative is to build the electronic superhighways as well as to develop the software and the hardware needed to bring to all groups in the population and all areas of the economy the benefits of the information age.

One feature of the information age is



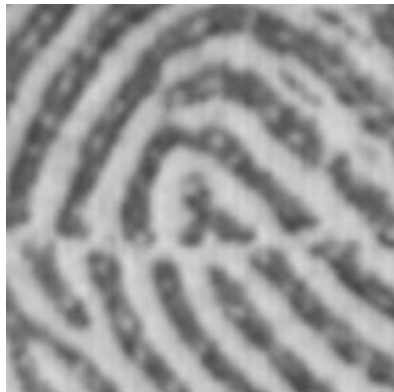
A digitized fingerprint image consisting of 768 by 768 8-bit pixels.



The corresponding data-compressed image resulting from application of the WSQ algorithm. The compression ratio is 21:1.



Detail of the center image after WSQ compression.



Detail of a digitized fingerprint image.



Detail of the center image after "JPEG" compression.

Figure 4. Wavelet Compression of Fingerprint Data

The FBI has over 29 million active cards in their criminal-fingerprint files. These cards are now being digitized at a spatial resolution of 500 pixels per inch and a gray-scale resolution of 8-bits. Each card yields about 10 megabytes of data, and the potential size of the entire database is thousands of terabytes. The FBI came to the Laboratory for help in organizing the data. The Laboratory's Computer Research and Applications Group collaborated with the FBI to develop the Wavelet/Scalar Quantization (WSQ) algorithm, which has been made a public standard for archival-quality compression of fingerprint images. (The algorithm involves a discrete wavelet transform decomposition into 64 frequency sub-bands followed by adaptive uniform scalar quantization and Huffman coding.) WSQ compression introduces some distortion in the image. The figures demonstrate, however, that the important features of the fingerprint, including branches and ends of ridges, are preserved. In contrast, the international "JPEG" image-compression standard, based on a cosine transform, is applied not to the original image as a whole but rather to square blocks into which the image has been divided. The image resulting from "JPEG" data-compression shows artifacts of this blocking procedure.



the rapid accumulation of very large sets of digital data—in the gigabyte and terabyte range. Such data sets are generated in a wide variety of contexts: large-scale scientific computation, medical procedures such as MRI and CAT scans, environmental surveillance and geoexploration by satellites, financial transactions, consumer profiling, law enforcement, and so on. The abilities to “mine” the data—to analyze them for meaningful correlations—and to compress the data for rapid communication and ease of storage are of increasing interest. Also needed are intuitive user interfaces for manipulation of those very large and complex data sets.

The Laboratory has initiated several data-mining projects that use sophisticated mathematical tools to solve practical problems of data analysis, storage, and transmission. One project has resulted in a new national standard, created in collaboration with the FBI, for compressing digital fingerprint data with little loss of information.

Figure 4 shows an original and a compressed fingerprint image. It also compares the method adopted by the FBI, known as the Wavelet/Scalar Quantization (WSQ) algorithm, with a traditional data-compression method. The compressed images will be transmitted between local law-enforcement agencies and the FBI to assist in retrieving past records of suspected criminals. Because the data have been

compressed by a factor of 20, the images can be transmitted in minutes rather than hours.

The WSQ algorithm transforms each image into a superposition of overlapping wavelets, localized functions that vanish outside a short domain (see Figure 5) in contrast to the sine and cosine functions of the standard Fourier transform, which oscillate without end. The discrete wavelet transform includes wavelets on many length scales and automatically produces a multiresolution representation of the image. Thus an image can be retrieved at whatever resolution is appropriate for a particular application.

The algorithm developed for the fingerprint data is also being used to create a multiresolution database for the efficient storage and retrieval of very large images. Aerial photographs of the Washington, D.C. area, supplied by the United States Geological Survey (USGS), were first digitized. Through the discrete wavelet transform, the many separate digital images were put together into a continuous image so that no seams are visible. The resulting multiresolution database is illustrated in Figure 6, which shows the area around the Lincoln Memorial in Washington, D.C. at seven decreasing levels of resolution. At the coarsest resolution (64 meters/pixel) the entire D.C. area can be displayed on a workstation monitor. At the finest resolution (1

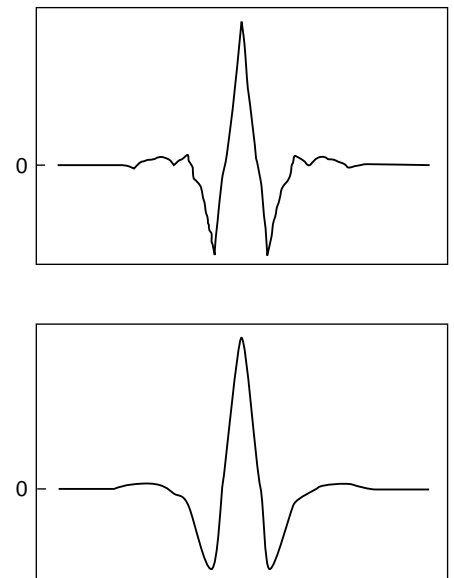


Figure 5. Mother Wavelets for the FBI Fingerprint Image Compression Standard

Shown here are the mother wavelets used in the WSQ algorithm for fingerprint data compression. The mother wavelets are translated and dilated to form the set of basis functions used to decompose and reconstruct the original image. The top wavelet is used for image decomposition and the bottom wavelet is used for image reconstruction.

meter/pixel) the user is able to distinguish features as small as an automobile. The USGS has an interest in making such data available for the

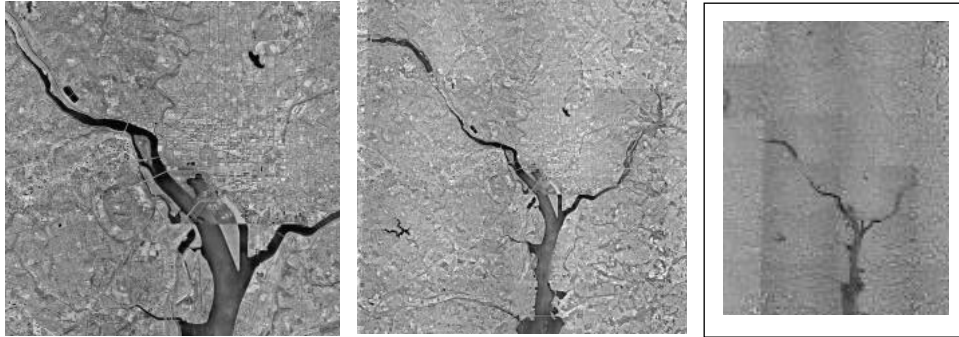


Figure 6. A Database of Seamless Multiresolution Images

The area around the Lincoln Memorial in Washington, D.C., is shown at seven resolutions. The digitized images were retrieved from a database constructed by using the discrete wavelet transform.

whole United States and disseminating the data on CD-ROM as well as on the electronic superhighways.

The multiresolution database project is part of a larger effort called Sunrise. The Sunrise project is a unique attempt to create an integrated software and hardware system that can handle many diverse applications of the kind envisioned for the nation's information superhighways—from access to very large databases, to secure electronic commercial transactions, to efficient communication of medical information in a national health-care system. The Sunrise strategy is to use a set of such diverse applications as a starting point for the development of software solutions, the elements of which are general enough to be used in many other applications.

The collaboration with radiologists at the National Jewish Center for Immunology and Respiratory Medicine on developing tools for telemedicine is illustrative of the Sunrise approach. The tools include a multimedia data management system that will display and analyze medical images, manage patient records, provide easy data entry, and facilitate generation of medical reports. The system is also designed to provide transparent access to medical information located anywhere on the information superhighway. Tools for interactive collaboration among physicians, efficient data compression, transmission, and storage, remote data storage

and retrieval, and automated data analysis are also being developed in the Sunrise approach to telemedicine (see "Concept Extraction" for a discussion of the quantitative analysis of LAM disease).

The Laboratory is going through dynamic change and nowhere is the change more visible than in the area of scientific computing, communications, and information systems. Because of the revolution in electronic communications, many people are doing things they never thought possible and in ways that could not have been anticipated (see "@XXX.lanl.gov" for a look into the future of research communication). Los Alamos is no longer focused solely on the high end of scientific computing for national security and basic research. We have become heavily involved in bringing advances in computing and information systems to all members of our Laboratory, to business and industry, and to the general public. We also expect that during the latter half of this decade advanced computer modeling and simulation will make increasingly direct and significant contributions to society. ■

Acknowledgements

We are grateful to the following people for their help with this article: Jonathan N. Bradley, Christopher M. Brislawn, John H. Cerutti, Salman Habib, Michael A. Riepe, David H. Sharp, Pablo Tamayo, and Bruce R. Wienke.

Further Reading

Yuefan Deng, James Glimm, and David H. Sharp. 1992. Perspectives on Parallel Computing. *Daedalus*, winter 1992 issue. Also published in *A New Era in Computation*, edited by N. Metropolis and Gian-Carlo Rota. MIT Press 1993.

N. Metropolis, J. Howlett, and Gian-Carlo Rota. 1980. *A History of Computing in the Twentieth Century*. Academic Press, Inc.

David W. Forslund is a Laboratory Fellow and Deputy Director of the Advanced Computing Laboratory. He is a theoretical plasma physicist who has contributed to controlled fusion and space plasma research and now specializes in distributed high-performance computing and NII technologies.

Charles A. Slocomb is the Deputy Division Director of the Computing, Information, and Communications Division. His primary interest is the future of high-performance computing and its application to scientific problems.

Ira A. Agins is a specialist staff member in the Computing, Information, and Communications Division. His avocation is computer history and particularly the history of computing at Los Alamos.

Collaborations with Industry on Parallel Computing

Bruce R. Wienke

The Computational Testbed for Industry (CTI) was established at the Laboratory in 1991 to provide U.S. industry with access to the computing environment at our Advanced Computing Laboratory and to the technical expertise of Los Alamos scientists and engineers. During this past year the CTI was designated officially as a Department of Energy User Facility. That designation affords us greater flexibility in establishing and implementing collaborative agreements with industry. The number of collaborations has been increasing steadily and will soon total about thirty. The seven projects described here are being established at the CTI through the new cooperative agreement between the DOE and Cray Research, Inc. under the auspices of the DOE's High Performance Parallel Processor program. The projects focus on developing scientific and commercial software for massively parallel processing.

Portability Tools for Massively Parallel Applications Development

Partners: Cray Research, Inc.; Thinking Machines Corporation

Goals: At present, software developed for one vendor's massively parallel computer system is not portable, that is, able to be run on other vendors' computers. The lack of portable programs has slowed the development of applications for every kind of massively parallel computer and the adoption of such computers by industry. This project will work toward removing that barrier by creating common programming conventions for massively parallel machines.

Massively-Parallel-Processing Performance-Measurement and Enhancement Tools

Partner: Cray Research, Inc.

Goals: Create a set of software tools to improve analysis of the system-level performance of massively parallel systems, to maximize their operating efficiency, and enhance the design of future systems. Plans include using this sophisticated automated toolkit to enhance the performance of applications developed in other projects under the cooperative agreement between the Department of Energy and Cray Research, Inc.

Lithology Characterization for Remediation of Underground Pollution

Partner: Schlumberger-Doll Research

Goals: Develop three-dimensional modeling software to cut the costs of characterizing and cleaning up underground environmental contamination. The software is intended for use by the petroleum and environmental industries on the next generation of massively parallel supercomputers.

Development of a General Reservoir Simulation for Massively Parallel Computers

Partners: Amoco Production Company; Cray Research, Inc.

Goals: Oil and gas exploration requires simulations of flow at several million points in reservoirs. Programmers have produced well-developed reservoir simulations for multi-processor vector supercomputers but not for massively parallel systems, so exploiting the potential of massively parallel computers is a high priority. The goal of this project is to adapt Amoco's field-tested reservoir-simulation software so that it performs efficiently on the massively parallel Cray T3D. The resulting program, which will allow much better management of reservoirs, will be made available to the entire petroleum industry.

Materials Modeling

Partner: Biosym Technologies Incorporated

Goals: In the competitive global marketplace for advanced materials, the traditional experimental approach to designing new materials needs to be complemented by materials modeling using high-performance computers. This project is aimed at creating powerful new visual-modeling software tools to improve casting and welding processes and to calculate the fracture properties of new materials designs, including composites.



At left: The Cray T3D was delivered to our Advanced Computing Laboratory in June 1994. It is a 128-processor machine that will be used primarily for collaborative research with industry. Below: Scientists and engineers from Cray Research, Inc. and Los Alamos at a dedication of the new machine.



Application of the Los Alamos National Laboratory Hydrocode Library (CFDLIB) to Problems in Oil Refining, Waste Remediation, Chemical Manufacturing, and Manufacturing Technology

Partners: Exxon Research and Engineering Company; General Motors Power Train Group; Rocket Research Company; Cray Research, Inc.

Goals: The speed and memory size of massively parallel systems will make it possible for U.S. industry to accurately model and improve the efficiency of chemical reactions that involve substances in more than one phase (solid, liquid, or gas). The project with Exxon will advance the simulation of multiphase reactors, which are heavily used in hydrocarbon refining, chemical manufacturing, gas conversion, and coal liquefaction and conversion. The goal of the General Motors project is to improve analysis of important foundry processes. One of the Rocket Research projects is aimed at improving small electric rockets used in satellite stations and has potential applications to microelectronics manufacturing and to neutralizing wastes in flue-gas emissions. Another Rocket Research project involves analysis of the performance, safety, and environmental impact of propellants used in automotive air bags and in fire-suppression systems of aircraft and other mass-transportation vehicles.

Massively Parallel Implicit Hydrodynamics on Dynamic Unstructured Grids with Applications to Chemically Reacting Flows and Groundwater Pollution Assessment and Remediation

Partners: Berea Incorporated; Cray Research, Inc.

Goals: Develop advanced software models to help U.S. industry better address problems involving combustion, pollution, and the treatment of contaminated groundwater and surface waters. These models could also be applied to designing engines, extracting more oil and gas from fields that have been drilled over, and assessing the structural integrity of buildings after a severe fire.

Collaborations with Industry on Parallel Computing

Bruce R. Wienke

The Computational Testbed for Industry (CTI) was established at the Laboratory in 1991 to provide U.S. industry with access to the computing environment at our Advanced Computing Laboratory and to the technical expertise of Los Alamos scientists and engineers. During this past year the CTI was designated officially as a Department of Energy User Facility. That designation affords us greater flexibility in establishing and implementing collaborative agreements with industry. The number of collaborations has been increasing steadily and will soon total about thirty. The seven projects described here are being established at the CTI through the new cooperative agreement between the DOE and Cray Research, Inc. under the auspices of the DOE's High Performance Parallel Processor program. The projects focus on developing scientific and commercial software for massively parallel processing.

Portability Tools for Massively Parallel Applications Development

Partners: Cray Research, Inc.; Thinking Machines Corporation

Goals: At present, software developed for one vendor's massively parallel computer system is not portable, that is, able to be run on other vendors' computers. The lack of portable programs has slowed the development of applications for every kind of massively parallel computer and the adoption of such computers by industry. This project will work toward removing that barrier by creating common programming conventions for massively parallel machines.

Massively-Parallel-Processing Performance-Measurement and Enhancement Tools

Partner: Cray Research, Inc.

Goals: Create a set of software tools to improve analysis of the system-level performance of massively parallel systems, to maximize their operating efficiency, and enhance the design of future systems. Plans include using this sophisticated automated toolkit to enhance the performance of applications developed in other projects under the cooperative agreement between the Department of Energy and Cray Research, Inc.

Lithology Characterization for Remediation of Underground Pollution

Partner: Schlumberger-Doll Research

Goals: Develop three-dimensional modeling software to cut the costs of characterizing and cleaning up underground environmental contamination. The software is intended for use by the petroleum and environmental industries on the next generation of massively parallel supercomputers.

Development of a General Reservoir Simulation for Massively Parallel Computers

Partners: Amoco Production Company; Cray Research, Inc.

Goals: Oil and gas exploration requires simulations of flow at several million points in reservoirs. Programmers have produced well-developed reservoir simulations for multi-processor vector supercomputers but not for massively parallel systems, so exploiting the potential of massively parallel computers is a high priority. The goal of this project is to adapt Amoco's field-tested reservoir-simulation software so that it performs efficiently on the massively parallel Cray T3D. The resulting program, which will allow much better management of reservoirs, will be made available to the entire petroleum industry.

Materials Modeling

Partner: Biosym Technologies Incorporated

Goals: In the competitive global marketplace for advanced materials, the traditional experimental approach to designing new materials needs to be complemented by materials modeling using high-performance computers. This project is aimed at creating powerful new visual-modeling software tools to improve casting and welding processes and to calculate the fracture properties of new materials designs, including composites.



At left: The Cray T3D was delivered to our Advanced Computing Laboratory in June 1994. It is a 128-processor machine that will be used primarily for collaborative research with industry. Below: Scientists and engineers from Cray Research, Inc. and Los Alamos at a dedication of the new machine.



Application of the Los Alamos National Laboratory Hydrocode Library (CFDLIB) to Problems in Oil Refining, Waste Remediation, Chemical Manufacturing, and Manufacturing Technology

Partners: Exxon Research and Engineering Company; General Motors Power Train Group; Rocket Research Company; Cray Research, Inc.

Goals: The speed and memory size of massively parallel systems will make it possible for U.S. industry to accurately model and improve the efficiency of chemical reactions that involve substances in more than one phase (solid, liquid, or gas). The project with Exxon will advance the simulation of multiphase reactors, which are heavily used in hydrocarbon refining, chemical manufacturing, gas conversion, and coal liquefaction and conversion. The goal of the General Motors project is to improve analysis of important foundry processes. One of the Rocket Research projects is aimed at improving small electric rockets used in satellite stations and has potential applications to microelectronics manufacturing and to neutralizing wastes in flue-gas emissions. Another Rocket Research project involves analysis of the performance, safety, and environmental impact of propellants used in automotive air bags and in fire-suppression systems of aircraft and other mass-transportation vehicles.

Massively Parallel Implicit Hydrodynamics on Dynamic Unstructured Grids with Applications to Chemically Reacting Flows and Groundwater Pollution Assessment and Remediation

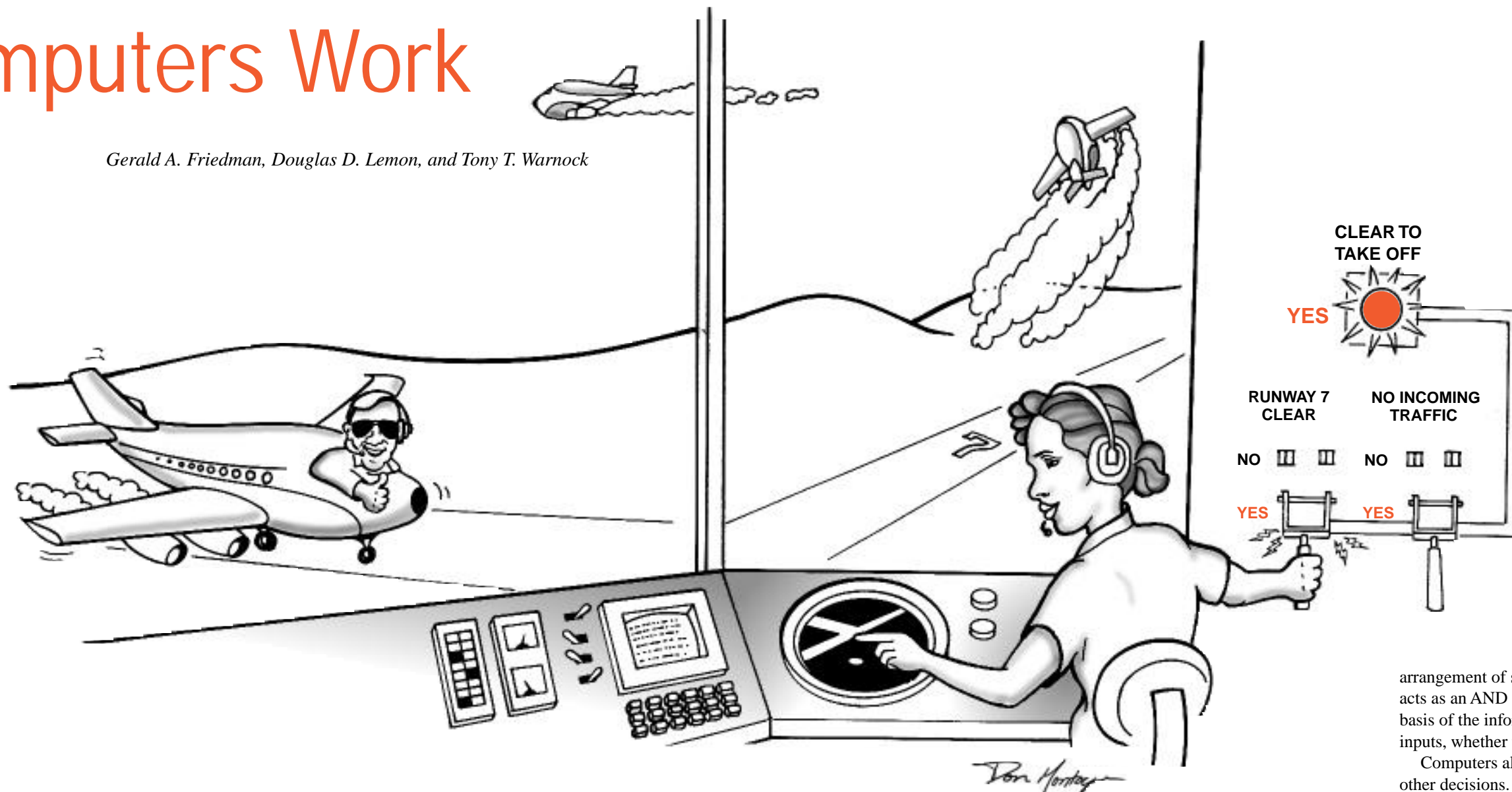
Partners: Berea Incorporated; Cray Research, Inc.

Goals: Develop advanced software models to help U.S. industry better address problems involving combustion, pollution, and the treatment of contaminated groundwater and surface waters. These models could also be applied to designing engines, extracting more oil and gas from fields that have been drilled over, and assessing the structural integrity of buildings after a severe fire.

How Computers Work

an introduction to serial, vector, and parallel computers

Gerald A. Friedman, Douglas D. Lemon, and Tony T. Warnock



Parallel computers provide the most powerful information-processing capabilities available. Like all digital computers, they represent and process information in ways that are based on simple ideas of Boolean logic. In this article we introduce those ideas and sketch the ways in which they are implemented in the elementary circuits of electronic computers. We then discuss the principles behind the high performance—in particular, the high speed—of vector and parallel supercomputers.

Digital Information, Boolean Logic, and Basic Operations

Suppose that an airplane is about to enter a runway for takeoff. The air-traffic controller must determine, with the aid of a computer, whether the plane can safely proceed onto the runway. Has the airplane that just landed cleared the runway? Are all incoming planes sufficiently far away? The answers to those questions—yes or no,

true or false—are information of a kind that can be managed by digital computers. The two possibilities are encoded within a computer by the two possible output states of electronic devices. For instance, “Yes” or “True” can be represented by an output signal on the order of 5 volts, and “No” or “False” by a near-zero voltage level. This simple code is the basis of the more complicat-

ed codes by which information is represented in computers.

Suppose that a light on the air-traffic controller’s console is to be lit if and only if (1) the runway is clear of other planes and (2) all incoming planes are at a safe distance. What is the mechanism by which an electronic computer “decides” whether to turn on the light? Each of the two conditions is represent-

ed by the voltage in a wire; when the condition is true the wire is at roughly 5 volts. (We imagine that the voltages are set by the controller using manual switches.) The wires are the inputs to a circuit called an AND gate. The output wire of the AND gate is at high voltage (to turn on the light) only when both input wires are at 5 volts, that is, when the following statement is true: The

runway is clear, *and* all incoming planes are at a safe distance. If the voltage that represents statement 1 is near zero (False), the output of the AND gate will also be at near zero voltage, so the light will not be lit. Likewise if the voltage representing statement 2 is low or if both voltages are low, the output will be at low voltage, and the light will not be lit. The simple

arrangement of switches drawn above acts as an AND gate that decides, on the basis of the information given by its inputs, whether to turn on the light.

Computers also use gates that make other decisions. For instance, OR gates give an output of True when either or both of two inputs is True. The decisions made by gates—whether an output is to be True or False depending on which inputs are True and which are False—are equivalent to the operations of Boolean logic (developed by George Boole, a nineteenth-century English mathematician). As the arithmetic operation of multiplication is specified by a multiplication table, Boolean operations (or gates) are specified by their

Figure 1. Examples of Truth Tables

NOT		AND			OR		
Input	Output	Input 1	Input 2		Input 1	Input 2	
			F	T		F	T
F	T	F	F	F	F	F	T
T	F	T	F	T	T	T	T
			Output			Output	

“truth tables,” which show all possible input combinations and the consequent outputs. Figure 1 shows truth tables for NOT, AND, and OR. Figure 2 shows examples of electronic circuits for NOT and AND gates.

The operations AND, OR, and NOT can be combined to make other Boolean operations by using the output of one operation as an input of another. For instance, computers frequently use the EQUAL operation, which gives a True output if and only if its two inputs are equal (both True or both False). That operation can be built from the gates already defined: Input 1 EQUAL Input 2 is equivalent to (Input 1 AND Input 2) OR (NOT Input 1 AND NOT Input 2). (The Boolean operations are combined in the same way as mathematical operations.) The EQUAL operation is used, for example, when a computer checks whether a user intends to delete a file. The computer might prompt the user, “If you’re sure you want to delete that file, type Y.” The character the user types is translated into a sequence of eight bits according to some code such as ASCII. If the sequence of bits representing the typed character is equal to the sequence of bits representing the letter Y, the output of the EQUAL circuitry activates the circuitry that deletes the file. In fact, circuits consisting only of the AND, OR, and NOT gates defined above can make any decision that depends only on which statements in a given set are true and which are false.*

In addition to processing characters and logical statements, computers must also process numbers. Two-state electronic devices represent numbers

*In principle, AND, OR, and NOT gates can all be built out of NAND gates (see Figure 2); they can also be built out of NOR gates (whose output is the opposite of that of an OR gate). Thus a computer could be built out of only one kind of gate.

through the use of the base-2, or binary, system—for instance, higher and lower voltages can indicate 1 and 0, respectively. A many-digit number is represented by the voltages in an ordered group of many such electronic devices. In fact, any kind of quantitative information can be represented by a series of higher and lower voltages. Furthermore, all of the operations of arithmetic can be performed (on binary numbers) by circuits made up of combinations of logic gates and other simple components. For example, the truth table of AND is a multiplication table if T is interpreted as 1 and F as 0.

Computers must also be able to store and retrieve information. Storage is performed by “memory” devices that can be set to maintain either a higher voltage or a lower voltage in their outputs and that, once set, maintain that voltage until they are reset. In our example of runway clearance at an airport, these devices were manual switches, but an electronic computer needs a memory containing a large number of two-state devices that can be set to either output state electronically. Such electronic circuits are called flip-flops. A flip-flop can be set to either state by the application of a voltage pulse to the proper input terminal, and it remains in that state until the application of another input voltage pulse. Thus each flip-flop “remembers” a single binary digit (abbreviated as “bit”), which is the smallest unit of quantitative information. Flip-flops are usually organized into cells, ordered arrays that each store a “word” of binary data (often 16, 32, or 64 bits long), which may encode a number, an alphabetic or other character, or quantitative information of any other kind.

The elements of electronic computers—electronic representation of infor-

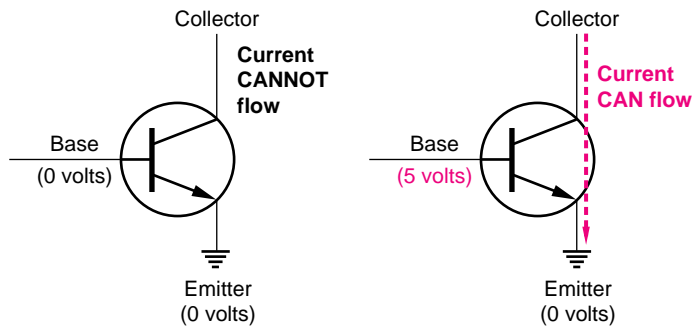
mation, logic gates, and flip-flops—provide the abilities to execute any Boolean operation and to read and write data. The English mathematician Alan Turing and others showed that if a machine has those abilities (and if it has sufficient memory and enough time is allowed), it can perform any known information-processing task that can be specified by an algorithm—a sequence of well-defined steps for solving a problem. An algorithm may include alternative actions depending on contingencies. A great many tasks can be specified by algorithms, as shown by the abilities of computers to perform them—such as controlling a satellite in orbit, trading stocks, and managing the text and figures for this magazine. Indeed, the limits of computers are unknown—though computers cannot yet comment insightfully on a poem, no one has conclusively ruled out that possibility.

Stored-Program Computers

Computers accomplish calculations and other tasks by executing “programs,” which are lists of instructions. Examples of instructions are fetching a number from memory, storing a number in memory, adding two numbers, and comparing two numbers or two characters. To “run” a program, the instruction list must first be loaded into memory in binary form. Then each instruction is read from memory and carried out by complicated circuits composed of the gates described above. Normally the instructions are performed in sequence. However, certain instructions allow different actions to be performed under different conditions. Depending on a specified input, these instructions cause execution either to

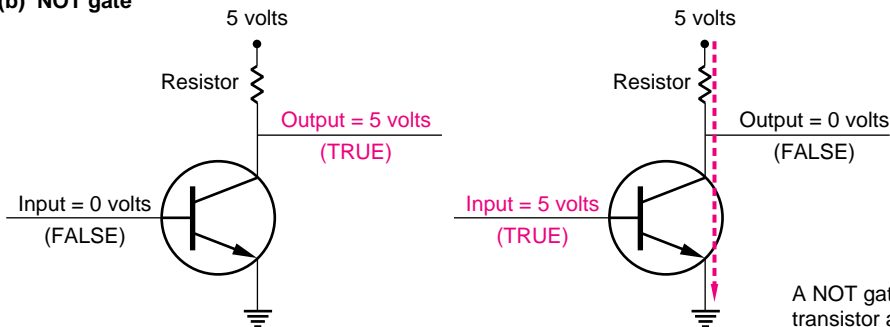
Figure 2. Computer Circuits

(a) Transistor



An NPN transistor can be put into either a conducting or a nonconducting state by the input voltage applied to it, so it can be used as the basic two-state switch in all the gates of an electronic computer. When an NPN transistor is used in a computer, the input is the voltage at the base and the output, which is controlled by the state of the transistor, is the voltage at the collector. (The emitter is grounded, or held at 0 volts). When the base is at a voltage near 5 volts, the transistor presents almost no resistance to the flow of electric current from the collector to the emitter, so the collector is effectively grounded. When the base is at a voltage near 0, the transistor blocks the flow of current between the emitter and the collector, so the voltage at the collector is determined by the other connections of the collector.

(b) NOT gate

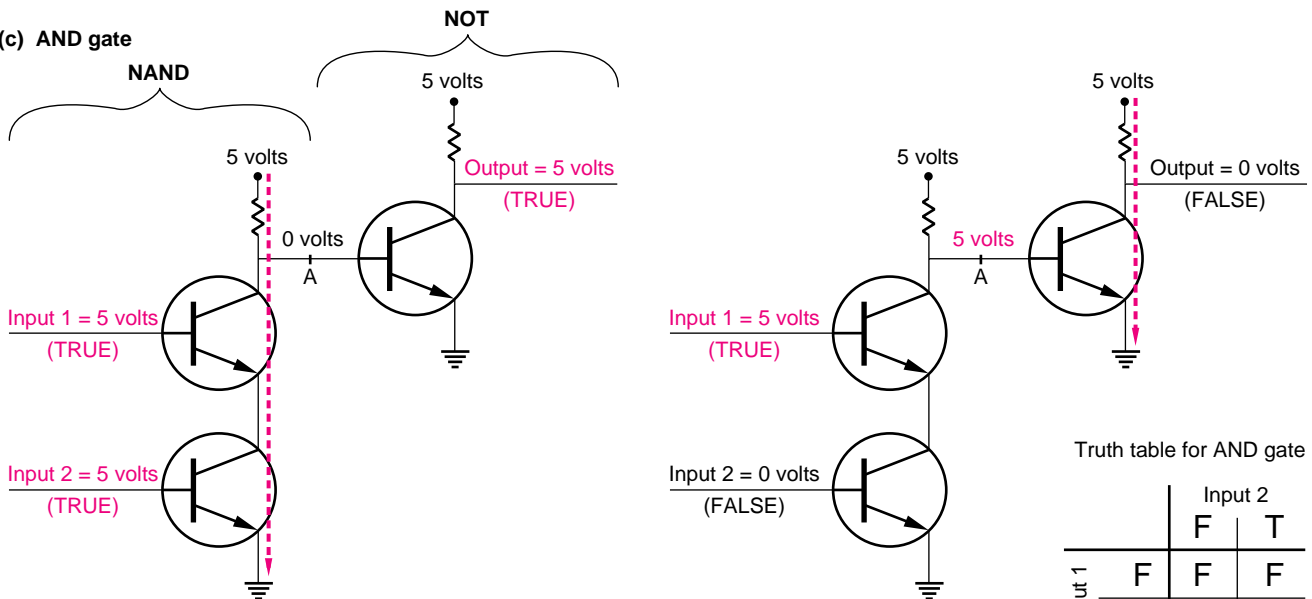


Truth table for NOT gate

Input	Output
F	T
T	F

A NOT gate, or inverter, can be made from an NPN transistor and a resistor. When the input is at 5 volts, the output (the collector voltage) is grounded and is thus fixed at 0 volts. When the input is at low voltage, the output is isolated from ground, no current flows through the resistor, and the output is consequently at 5 volts.

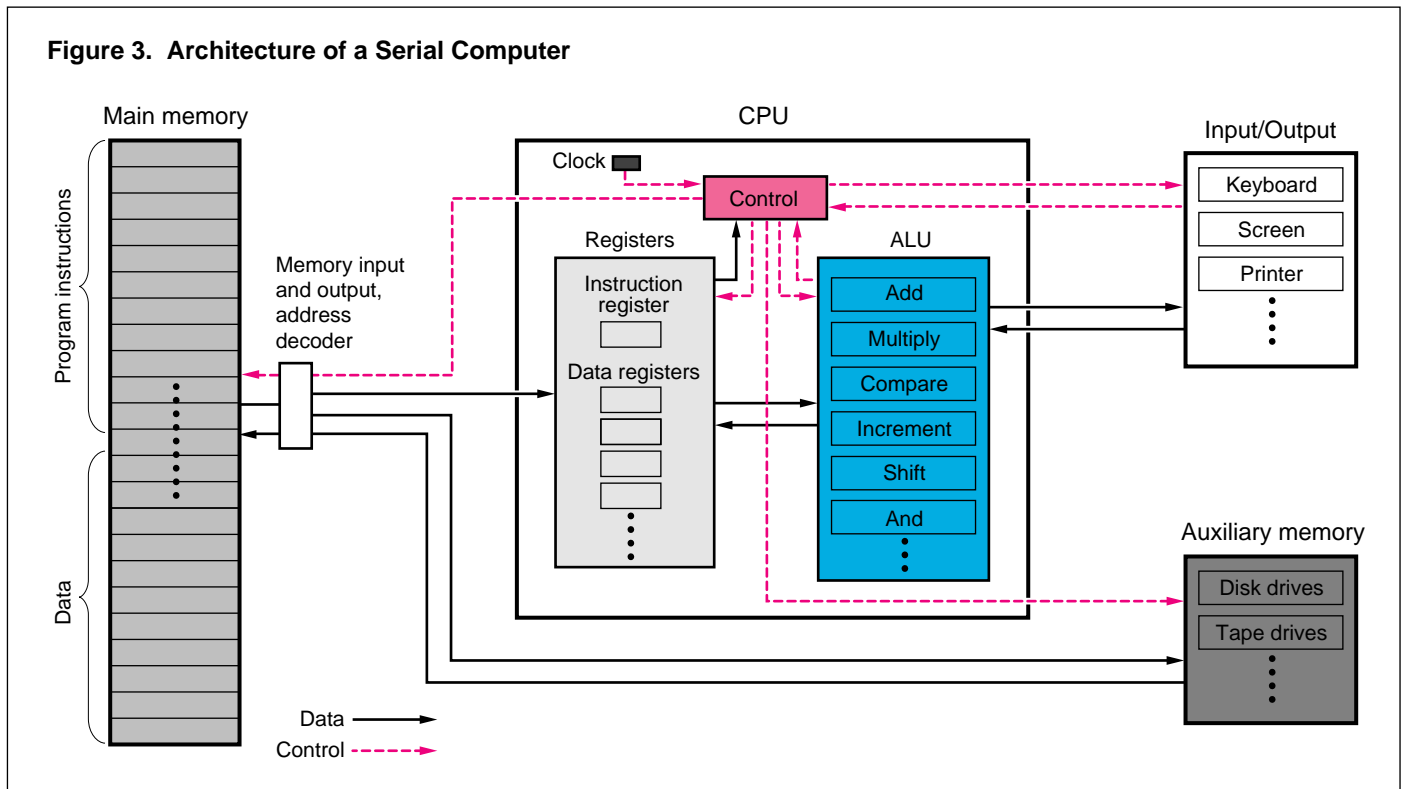
(c) AND gate



Truth table for AND gate

Input 1	Input 2		Output
	F	T	
F	F	F	F
T	F	T	T

An AND gate can be made from NPN transistors and resistors. The part of the circuit to the left of point A is actually a NAND gate; that is, the voltage at point A is given by the operation NOT(input 1 AND input 2). The NAND gate operates by a principle similar to that of the NOT gate, and the voltage at point A is inverted by the NOT gate to the right. When both inputs are at 5 volts, point A is grounded (False), so the output of the AND gate is True. When either input is near 0 volts, point A is isolated from ground and thus at 5 volts (True), so the output of the AND gate is False.



“jump” to a specified instruction elsewhere in the program or to continue with the next instruction in sequence.

Serial Computers

The simplest electronic computers are serial computers—those that perform one instruction at a time. Figure 3 shows a typical example of the architecture, or overall design, of a modern serial computer. Note that the architectures of computers vary enormously on all levels, from the choice and configuration of the large-scale components down to the choice of binary code used to represent numbers. Every illustration in this article shows an example chosen from among many possibilities.

The major components of a serial computer appear in Figure 3. The central processing unit (CPU) is the heart

of the computer. The CPU in a modern serial computer often consists of a single integrated-circuit chip. It contains the arithmetic-logic unit (ALU), the part of the computer that actually computes. The ALU consists of circuits that perform various elementary processes such as addition, multiplication, comparison of numbers, and Boolean operations. Such operations are performed on data that are in registers—very quickly accessible memory cells located within the CPU. The control unit contains circuits that direct the order and timing of the operations by “gating,” or opening and closing electronic pathways among the ALU, the memory, and the input/output units. A typical calculation involves loading data from main memory into registers, processing of those data by the ALU, and storage of the results in main memory. Note that the CPU typically includes a

clock, a device that synchronizes operations by emitting electric pulses at regular intervals.

Main memory stores information that can be read and written by other parts of the computer. It consists of memory cells that are packed together densely on a device consisting of one or more semiconductor chips; a single chip can store many millions of bits of information. Each cell is labeled by a numerical address, and other parts of the computer can read from or write to any cell by sending its address and a read or write signal to the memory device. Because the memory cells of the device can be accessed in any order, main memory is called random-access memory, or RAM. Main memory is “volatile;” that is, the stored information is lost when the power is switched off. Therefore main memory typically stores information that the CPU will

use in the short term, such as the program being executed and the data that the program processes and generates.

The computer contains a very large number of connections between its main components and within the CPU. The connections are conducting paths that make the voltage at the input of a circuit equal to the voltage at the output of another. The lines in Figure 3 indicate schematically whether one part of the computer is connected to another part. Solid lines indicate data connections; dashed lines indicate control connections. The connections consist of multiple conducting paths or wires. For instance, the connection between main memory and the CPU includes separate pathways for data transfers, addresses, read and write signals, and other purposes. Furthermore, the data path must have a number of wires equal to the number of bits that must be transferred. Thus in a computer whose words consist of 16 bits, a data connection between main memory and the CPU consists of 16 wires.

Auxiliary memory refers to non-volatile storage devices such as disk and tape drives. Auxiliary memory is generally much more capacious than main memory (and less expensive per bit of capacity). Present auxiliary memories have capacities of up to millions or billions of words. However, the access (read/write) times for auxiliary memory are greater than those for main memory. Therefore auxiliary memory is generally used to store programs and data while they are not in use.

Input and output devices communicate data between the main memory and users or other computers, translating the data between bit sequences used by the computer and forms understandable to people (or other computers). Probably the most common input device is the keyboard, and the most common output devices are the printer and

the monitor. There are many other more- or less-specialized input-output devices, such as network connections, modems, mice, scanners, loudspeakers, and so forth. Some auxiliary-memory devices can be used for input/output—for instance when one computer writes onto a floppy disk that another computer will read.

To show how a serial computer performs a calculation, we will take as an example the calculation of the salary raises r of a company's 800 employees. Each raise is given by the product of an employee's present salary p and an adjustment factor a previously calculated on the basis of the employee's performance. In our example we use the common technique of storing the values of p in the computer as a one-dimensional array, or vector, $p(i)$; the value of p for the first employee is called $p(1)$, and so forth. When the program runs, the elements of $p(i)$ are stored in consecutive memory cells. The values of a are stored similarly as elements of a vector $a(i)$. Thus the calculation can be written as $r(i) = p(i) * a(i)$ for all i .

Figure 4 shows how an exemplary serial computer performs a multiplication. Since the auxiliary memory and input-output devices are not involved in this calculation, they are omitted from the figure. The program for the raise calculation has been loaded into main memory; each instruction is stored as a 16-bit number. The figure shows the location of one instruction, which is represented by the four digits 3123. The first digit in the instruction is interpreted by our example control unit as an operation, the next two digits are the numbers of the two registers containing the operands, and the final digit is the number of the register where the result is to be placed. Here 3 is the code for MULTIPLY; then the instruction 3123 means, "Multiply the number in register

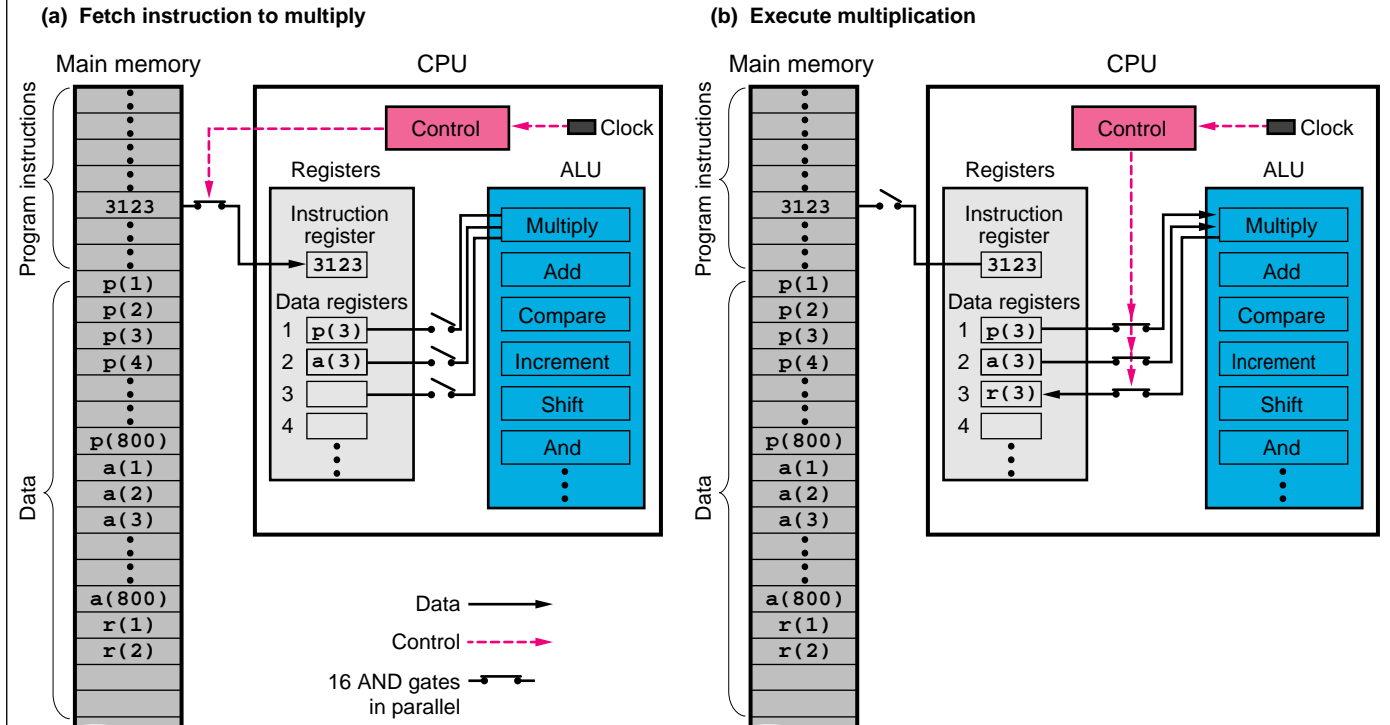
1 by the number in register 2 and place the result in register 3."* The $p(i)$ and $a(i)$ arrays are also in main memory. In this example, each of the first two elements of $p(i)$ has already been multiplied by the corresponding element of $a(i)$, and the resulting $r(1)$ and $r(2)$ have been stored in main memory. (The cell where $r(3)$ will be stored is still blank.) The numbers in the registers— $p(3)$ and $a(3)$ —were loaded by the execution of the previous instructions.

The computer in this example performs instructions in two stages: First it fetches an instruction from main memory, then it executes the instruction. Figure 4a shows the fetching process. We begin following the process just after the instruction to load $a(3)$ has been executed. As always after an instruction has been executed, the control unit is in a state such that a voltage impulse from the clock causes it to fetch the next instruction. (The control unit was initially put in that state by the "run" command that began the execution of a program.) Also, the address of the multiply instruction is in the instruction-address register (not shown). At the clock signal, the instruction, stored as the number 3123, is fetched from main memory into the instruction register. The bottom part of Figure 4a illustrates how the outputs from the control unit transfer, or gate, the instruction from main memory to the instruction register. In general, gating is the control unit's primary means of controlling the operations.

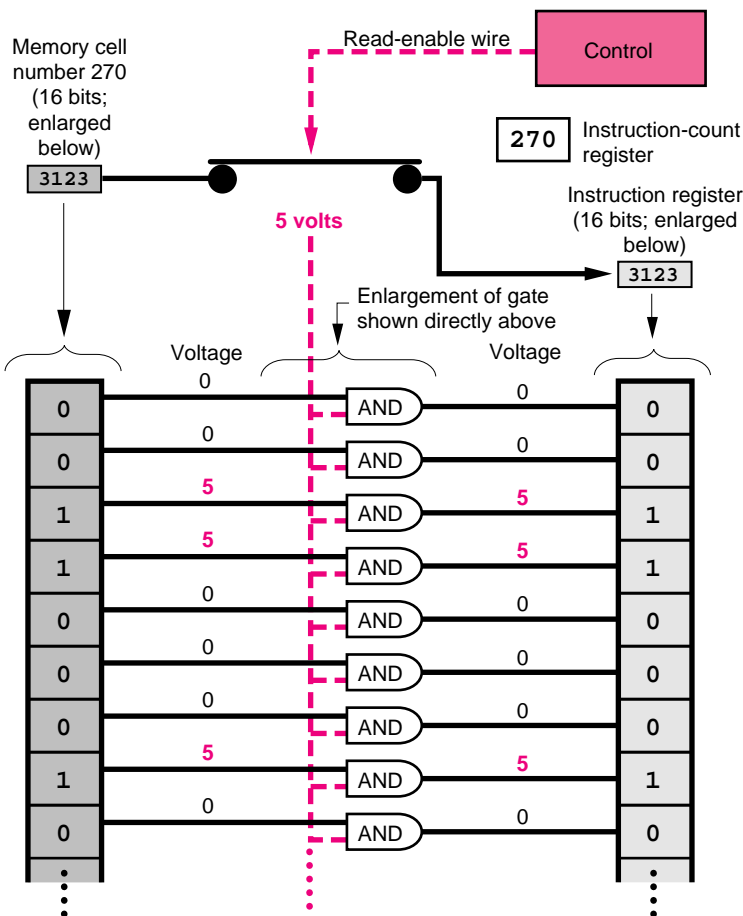
At the next clock signal, the instruction is gated into circuits in the control unit. Figure 4b shows that the resulting outputs from the control unit gate the

*In this example, each digit stands for one of the 16 four-bit numbers, and is therefore to be interpreted in base 16. The instruction represented by the base-16 number 3123 would be stored as the binary representation of that number: 0011 0001 0010 0011.

Figure 4 . Execution of One Instruction



Detail of gating shown in (a) above



Shown at left is the gating of an instruction, which consists of a single word, from a cell in main memory to the instruction register. A number of other connections controlled by AND gates are involved in fetching the instruction; for clarity they are not shown, but they are described in the paragraph below. The instruction-count register specifies that the word to be copied is in memory cell number 270. We assume the computer was designed to use words consisting of 16 bits; therefore the memory cell and the instruction register each consist of 16 flip-flops. The output of each flip-flop of the memory cell is connected to one input of an AND gate, and the read-enable wire from the control unit is connected to the other input of each AND gate. When the read-enable wire (red dashed line) is set by the control unit to 5 volts as shown, the output wire of each AND gate is at the same voltage as its input from memory. Therefore the output of each flip-flop of the instruction register is set to 5 volts or 0 volts depending on whether the bit stored in the corresponding flip-flop in main memory is 1 or 0. Thus the control unit "gates" the 16 bits of the word from the memory cell to the instruction register. However, when the read-enable wire is at 0 volts, all the inputs to the instruction register are at 0 volts. This set of AND gates might be compared to a starting gate at a racetrack, which either holds all the horses in place or lets them all run at once.

The appropriate memory cell is chosen by use of the circuit called the address decoder. The control unit gates the address of the desired instruction from the instruction-count register to the address decoder which, in combination with a signal from the control unit to the entire memory, turns on the read-enable wire of the memory cell at that address and no other. The output of the cell goes to the memory-output pathway (which along with the address decoder and the memory-input pathway was shown adjacent to the main memory in Figure 3). Yet another signal gates the contents of the memory-output pathway to the instruction register.

numbers in registers 1 and 2 to the multiplier and gate the output of the multiplier, which is the product of the two numbers, into register 3. The execution of the multiplication takes several clock cycles.

Other instructions, perhaps seven or eight, must be executed as part of the routine that calculates each employee's raise. Before the multiplication the memory addresses of $p(i)$, $a(i)$, and $r(i)$ are calculated on the basis of the value of i . As we have mentioned, the factors $p(i)$ and $a(i)$ are gated into registers. This is the most time-consuming step; loading a word from memory may take 20 clock cycles. After the multiplication, the number in register 3 is stored in memory at the address for $r(i)$. Finally, the value of i is incremented and the result is compared with 800 to determine whether the calculation is finished. If not, the control unit jumps to the beginning of the routine (by writing the address of the first instruction of the multiplication routine into the instruction-address register) and repeats the routine for the next value of i . The total time to perform the routine for one value of i may be about 30 clock cycles.

Typical processors contain circuits to perform all these operations. The specifics of the elementary instructions, however, vary greatly among different computers, and the choice of "instruction set" is an important part of computer architecture.

A computer's instruction set is also called its assembly language. Users can write programs in assembly language, one encoded instruction at a time. However, the coding is quite tedious, and the result is difficult for a human being to read. Moreover, since instruction sets differ among computers, an assembly-language program written for one type of computer cannot be "understood" by any other. Therefore,

many high-level languages have been developed. They consist of instructions that combine several assembly-language instructions in ways that are convenient for humans. For instance, when written in the popular language FORTRAN, the instructions (or code) for the raise calculation above might look like this:

```
do i = 1, 800
    r(i) = p(i)*a(i)
end do
```

Those lines of FORTRAN code instruct the computer to perform the statement between the `do` and the `end do` for all values of i from 1 to 800. Programs called compilers, assemblers, and loaders translate the high-level code into binary instructions and assign memory addresses to variables. Not only is the FORTRAN code easier to read than assembly code, but it is also portable; that is, it can be executed by any computer that has a FORTRAN compiler. (Complicated programs may not be perfectly portable—minor adjustments may be required to run them on computers other than the original.) All the advantages of high-level languages are counterbalanced by some cost in execution speed, so sometimes when speed is the highest priority, programmers write in assembly language.

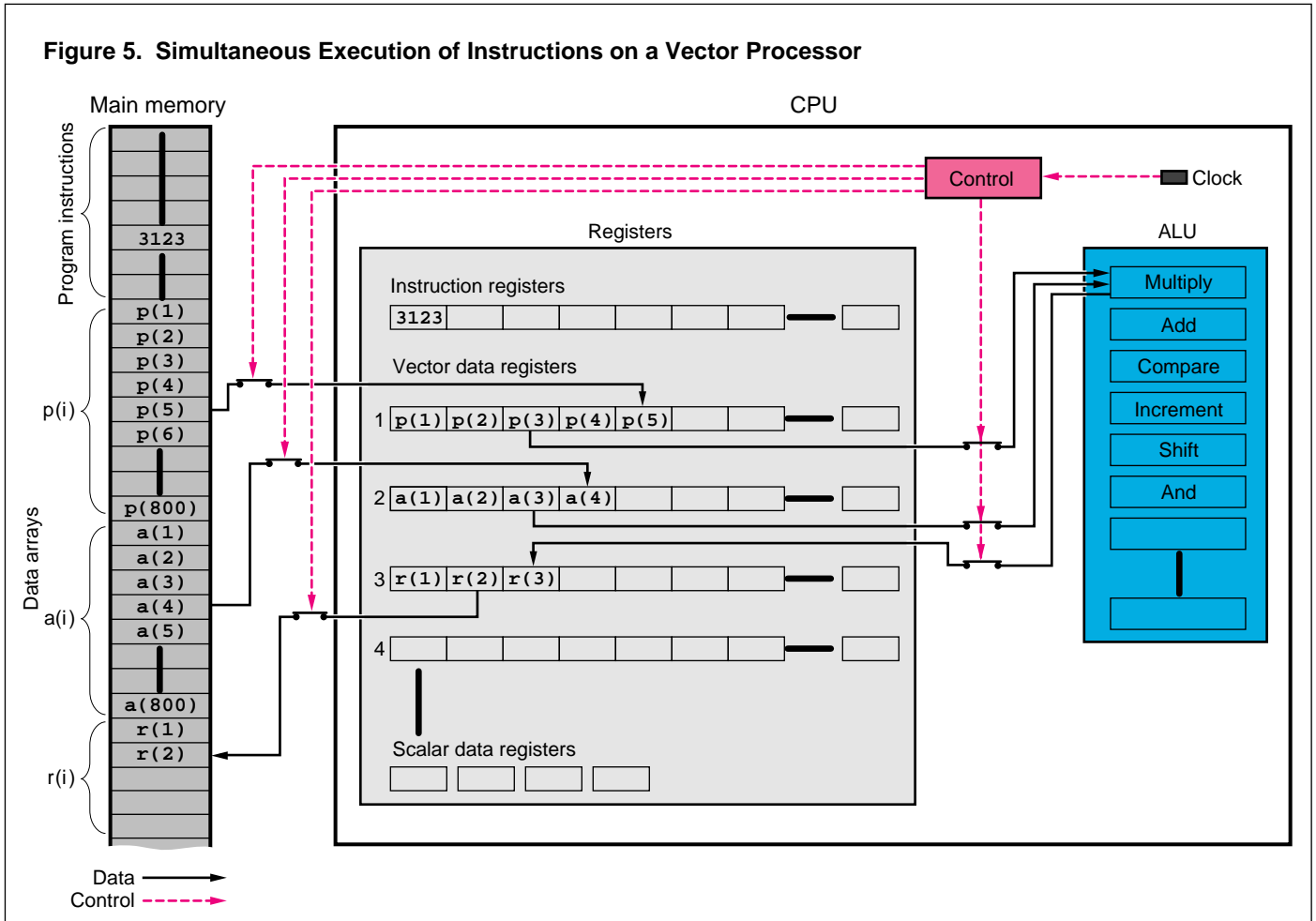
Speed is not always the highest priority, but it is always important. Speed is continually being increased by the development of faster and smaller gates and flip-flops. (Smaller components can be packed on a chip in larger numbers so that there are more of the fast connections within chips and fewer of the slower connections between chips.) Integrated-circuit technology has reached the point where mass-produced CPU and memory chips are fast enough to serve as components in the fastest computers used for large calculations in science and engineering.

To take advantage of high-speed components, manufacturers use architectures designed for high speed. For instance, the main memory must be large so that it can store all the data for large problems without requiring transfers of data to and from auxiliary memory during the calculation. (Such transfers are relatively slow.) Also, the bandwidth, or capacity, of the connections must be increased by adding wires, so that the connections can transfer more data in a given amount of time. These features provide higher performance, but serial computers have the essential limitation that instructions can be performed only one at a time. Therefore supercomputers—the fastest (and most expensive) computers—usually have architectures that allow the performance of several instructions at once. There are two such architectures: vector and parallel.

Vector Computers

Vector computers are based on processors that execute calculations on the elements of a vector in assembly-line fashion. As illustrated in Figure 5, a vector processor contains five to ten "functional units" that can act simultaneously, each carrying out an instruction. Functional units are thus analogous to the stations on an assembly line. The functional units include various circuits on the ALU. Three additional functional units are the separate connections between main memory and registers that allow the simultaneous loading of two numbers into registers and storing of one number from a register. (These connections are an example of high-bandwidth connections that speed up transfers of data.) Another distinctive feature of vector processors is that they contain several vector registers. The vector registers shown in Fig-

Figure 5. Simultaneous Execution of Instructions on a Vector Processor



ure 5 can contain 64 elements of a vector at once.

We use the calculation of raises as our example again. As shown in Figure 5, the calculation is set up in “assembly-line” fashion so that at any time each functional unit is working on a different element of the result r . At the stage shown, the first element of r has been completely processed and stored. The second element is being stored, the multiplication for the third element is being performed, and data for the fourth and fifth elements are being loaded from RAM into elements of vector registers. When 64 elements of r (enough to fill a vector register)

have been calculated, the process repeats to compute the 65th through 128th elements, and so forth until all 800 elements have been computed.

After the first element of r is stored, another element comes off the line in every clock cycle—just as cars may come off an assembly line every hour even though assembling a single car takes days of work. Since a serial calculation of a single element requires perhaps 30 clock cycles, a vector processor that performs those instructions simultaneously reduces the calculation time by a factor of roughly 30. Calculations that use more of the processor’s functional units are sped up

still more. Furthermore, each instruction is fetched only once and executed many times, so the speedup is greater still. (In a typical serial computer, fetching each instruction requires a clock cycle or more.) Some “overhead” time is needed to set up the assembly line, but that time can often be reduced or eliminated because functional units not used in one assembly line can set up the next one. Despite a slight additional overhead at the beginning and end of the calculation when some functional units on the assembly line are idle, speedups of factors of hundreds are possible in favorable cases.

However, many calculations cannot be sped up by vector processing because they cannot be “vectorized,” or put in the form of identical operations on elements of vectors so that each functional unit has a predictable task on the “assembly line.” The number of elements of the vectors must be at least comparable to the number of elements of the vector register so that the time saved by performing the calculation on an assembly line more than compensates for the overhead. Also, the calculations on different components should be independent of each other so that they can proceed simultaneously. An example that does not satisfy that condition is the calculation of the sum of many terms, $\sum_i x(i)$. If the calculation is performed by the straightforward method of adding each element in turn to the subtotal, it cannot be vectorized because the subtotal of the first i elements must be completely calculated before the addition of $x(i+1)$ can begin. (Ingenious methods for vectorizing sums of many terms have been invented, but nonetheless vector computers perform sums more slowly than such calculations as the multiplication shown in Figure 5.) Calculations that vector computers can perform particularly fast include linear algebra, Fourier and other integral transforms, and other calculations used in such scientific applications as finite-difference and finite-element simulations of fluid flow. Even when a calculation cannot be vectorized, vector computers often provide some speedup if the calculation involves large amounts of data, because one may be able to vectorize the loading of the data from main memory and the storing of the results.

Vector computers have been made relatively easy to program. In fact, programs written for a serial computer can usually run on vector computers, although to take full advantage of vector

processing, the programs often must be rewritten. However, techniques for doing so are well known. Furthermore, much of the necessary rewriting is now automated; that is, the programmer can write code as if for a serial machine, and the compiler will translate the program to assembly code optimized for the vector processor.

Parallel Computers

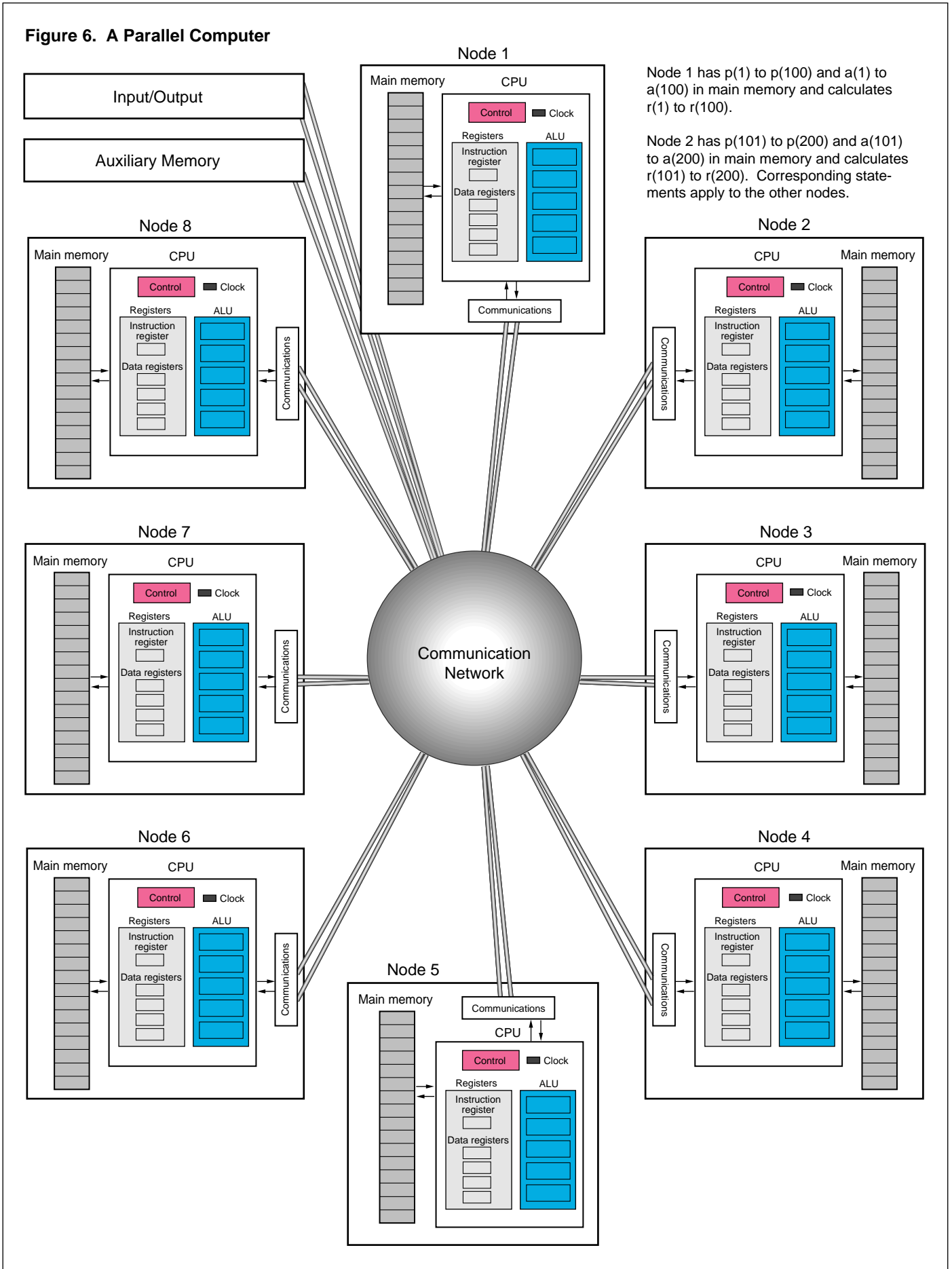
Parallel computers consist of a number of identical units that contain CPUs and essentially function as serial computers. Those units, called nodes, are connected to one another and simultaneously perform more or less the same calculation on different parts of the data. For example, Figure 6 shows an eight-node parallel computer calculating the salary raises of our earlier example. Each processor is connected to its own main memory. (This arrangement is called distributed memory.) The nodes are connected to each other by a high-bandwidth communication network. Each of the eight nodes stores and processes only one-eighth of the data—node 1 stores $p(1)$ through $p(100)$ and $a(1)$ through $a(100)$, and it is calculating $r(1)$ through $r(100)$. The maximum possible increase in speed would be a factor equal to the number of nodes. If that speedup can be attained for a given calculation, the calculation is said to be scalable. In practice, the speedup is somewhat less, because time must be spent in communicating the data between nodes and in coordinating the functions of the nodes. When a calculation can be made nearly scalable, it is well-suited for massively parallel computers—those that have hundreds of processors.

Parallel computers have the great advantage over vector computers that they can be built from off-the-shelf CPUs

and other components made for serial computers. Since vector computers require special-purpose vector processors, parallel computers can offer higher maximum speed (in terms of operations per second) than vector computers of the same price. However, running parallel computers at their maximum speed is more difficult than running vector computers at their maximum speed. The two basic problems of parallel computers are the coordination of the processors and their access to data.

A simple hardware method of making data available to all processors would be to change the architecture shown in Figure 6 by giving the computer a shared memory—one extremely large memory to which all the processors have equal access. However, for technical reasons the number of data and control connections to a shared memory increases faster than the number of processors, so making massively parallel shared-memory computers is impractical at present. Therefore most massively parallel computers, like the computers sketched in Figure 6, have distributed memories, and each processor can retrieve data directly from its own memory only. Because reasons of cost require connections between the nodes of a distributed-memory computer to have less bandwidth than those between a processor and its own main memory, data transfers between nodes are relatively slow. An important problem in programming such computers is minimizing the time spent in transferring data from the memory of the processor that stores them to the memory of the processor that needs them. Several solutions to this problem have been tried. The choice of solutions presents computer buyers with a trade-off: A computer that handles the communication and control automatically and thus hides communication and control problems from the

Figure 6. A Parallel Computer



user is easier to program than one that requires the user to solve those problems. However, automation is relatively inflexible, so such a computer can speed up fewer kinds of programs than one that requires the user to do more programming work.

The earliest approach to solving the communication problem is particularly automatic and rigid. It is known as the single-instruction/multiple-data, or SIMD, architecture. In SIMD computers, all processors execute the same instructions in synchrony on different parts of the data set. On a SIMD machine, the control of the processors and communication between them is straightforward. Programmers can write code for SIMD computers in a style identical to that for serial computers, and each processor executes a copy of the code. Since the problems are divided in a predictable way, data can be distributed to the processors that need them without any need for complex calculations. However, only a limited set of problems can be approached so simply. In fact, most calculations suitable for SIMD computers are equally suitable for vector computers.

The other common parallel architecture is MIMD (multiple-instruction/multiple-data), which is being used for most currently available parallel computers. Each processor in a MIMD computer functions independently of the other processors. Because MIMD systems have greater flexibility than vector or SIMD systems, they work well on more kinds of programs: for instance, databases, simulations of many interacting bodies (see "State-of-the-Art Parallel Computing" and "A Fast Tree Code for Many-Body Problems"), and Monte Carlo simulations (see "A Monte Carlo Code for Particle Transport"). However, interprocessor communication in MIMD systems is more complicated than in SIMD systems.

The different approaches to communication in MIMD machines include different programming models. (A computer's programming model is the structure of the programs it is designed to run.) Two programming models that offer relative ease of programming but limited versatility are data parallelism and distributed computing. Data parallelism is similar to SIMD in that each processor performs the same routine, but differs in that each processor can spend a different amount of time on the routine. High-level languages have been devised to automate most of the communication in data-parallel computation. At the other extreme of processor synchrony is distributed computing, in which the processors are nearly independent. Distributed computing is particularly useful in Monte Carlo calculations; then each processor can perform simulations independently of all the others.

The most versatile programming model, between data parallelism and distributed computing in the degree of processor independence, is message-passing. This model effects communication by having the processors send messages to each other as needed to request or transmit data. Thus communication can be performed more flexibly than in data-parallel computers, where communication must fit into a fixed model. However, programs that call for message passing must include specific instructions for the messages. Therefore programmers often need to understand the details of how the particular computers they use handle communication. Programmers are making progress in applying messages-passing methods, as described in several articles in this issue. We should note that some massively parallel computers allow the user to choose or combine programming models to suit each problem. The CM-5 Connection Machine supports

both the data-parallel and the message-passing models. The article "State-of-the-Art Parallel Computing" describes the Laboratory's CM-5, including the control of and communication among the processors.

Finally, we should note that there is no reason not to combine vector and parallel architectures. Vector computers have long incorporated up to sixteen processors with a shared memory. Furthermore, one modern supercomputer, the CM-5, is a massively parallel computer in which each node contains one or more vector processors in addition to its main CPU. Computers of these "hybrid" types dominate the supercomputer market. Making predictions about supercomputers is very risky because of the rapid development of hardware, software, and architectures, but we can expect that for some time the fastest supercomputers will probably be massively parallel computers and vector-parallel hybrids. ■

Acknowledgement

We are grateful to Randy E. Michelsen for highly valuable discussions.

Further Reading

Caxton C. Foster and Thea Iberall. 1985. *Computer Architecture*. Van Nostrand Reinhold Company.

Alan Freedman. 1991. *The Computer Glossary: The Complete Illustrated Desk Reference*, fifth edition. The Computer Language Company Inc.

Dharma P. Agrawal, editor. 1986. *Tutorial: Advanced Computer Architecture*. IEEE Computer Society Press.

Anthony Ralston, editor. 1976. *Encyclopedia of Computer Science*. Petrocelli/Charter.

Glenn Zorpette, editor. 1992. *Teraflops Galore*. September 1992 issue of *IEEE Spectrum*.

H I P P I

the first standard for high-performance networking

Stephen C. Tenbrink and Donald E. Tolmie

In the late 1980s, Laboratory researchers looked at the links in their local-area network—specifically, the connections between supercomputers, file storage, and other devices inside the machine room—and saw potential bottlenecks. The network had a maximum data-transmission rate of 50 million bits per second, one of the highest in the country, but numerical simulations on the fastest supercomputers were generating increasingly large data files. The time would come when simply moving the files would take almost as much time as generating them. Visualizing the data presented another bottleneck. For example, a researcher might want to simulate the formation and merging of eddies in a turbulent fluid and then view the simulation at a rate that the eye interprets as smooth, continuous motion—about 30 frames per second. To show 30 frames each second on a high-quality, 24-bit color monitor with a 1024-by-1024 pixel display requires a data-transmission rate of about 750 million bits per second—15 times higher than what the network could support in 1987.

The solution to that problem required more than just a system for high-speed data transmission. In 1987 there was already a sufficiently high-speed communication link—the proprietary Cray HSX interface, which operated at approximately 850 million bits per second. However, the Laboratory (and other supercomputing facilities) wanted to be able to transmit data freely among computers, monitors, storage devices, and so forth. A data-transmission system for that purpose would have to meet two re-

quirements that the HSX interface did not meet. First, the equipment for the system had to be nationally standardized so that any manufacturer could give its devices high-speed ports compatible with the new data link. Second, since connecting every device to every other device with a separate cable was impractical, the new system had to be a network in which switches would connect any device to any other as needed. Therefore Laboratory researchers decided to develop and standardize a new generation of networks that would transmit data at about a gigabit (one billion bits) per second.

National standards are developed by committees of volunteers under the auspices of standards organizations such as the American National Standards Institute (ANSI). In 1987 Laboratory personnel presented their plans for a gigabit-network national standard to an ANSI group working on high-speed data transmission. That committee had been considering a more modest standard, specifying transmission speeds of 100 to 200 million bits per second, to be used in permanent connections between computers and data storage. The committee's reception of the Los Alamos proposal was decidedly cool—the members referred to advocates of gigabit speeds as the “lunatic fringe.”

Two months later, proponents of gigabit networks from both the Laboratory and industry joined the committee. Within a year that committee produced the first draft of the standards for the new gigabit network, now called HIPPI, the high-performance, parallel interface. As soon as word got out that the new

standard was on the way, vendors began to build components for the gigabit network. The first commercial products appeared in 1988: a HIPPI port for the IBM 3090 and a fiber-optic extender for HIPPI networks. In the same year, the Laboratory built a prototype of a HIPPI switch. By 1989, the first commercial HIPPI switches were marketed, and HIPPI ports on computers made by IBM, Cray Research, and the Thinking Machines Corporation were being developed.

The ANSI committee developed the HIPPI standards rapidly by basing them as much as possible on off-the-shelf technology. In November 1991, three years after the initial draft was delivered to ANSI, HIPPI became the first national standard for gigabit-per-second data transmission. (More precisely, the HIPPI standard gives specifications for transmitting data at two speeds: either 800 million bits per second—the minimum speed for a high-resolution, 30-frame-per-second movie—or twice that speed, 1.6 billion bits per second.) The HIPPI standard was adopted in a remarkably short time—a more typical time for acceptance of a new network standard is five to ten years. Today, HIPPI is the interface of choice for gigabit-per-second transmissions on most supercomputers and some high-performance workstations because it was the first standard at those speeds and because the technology is mature and stable. For instance, the Laboratory now has one HIPPI network that is in regular use and another that serves as a testbed for HIPPI hardware and software development. Newer gigabit network stan-

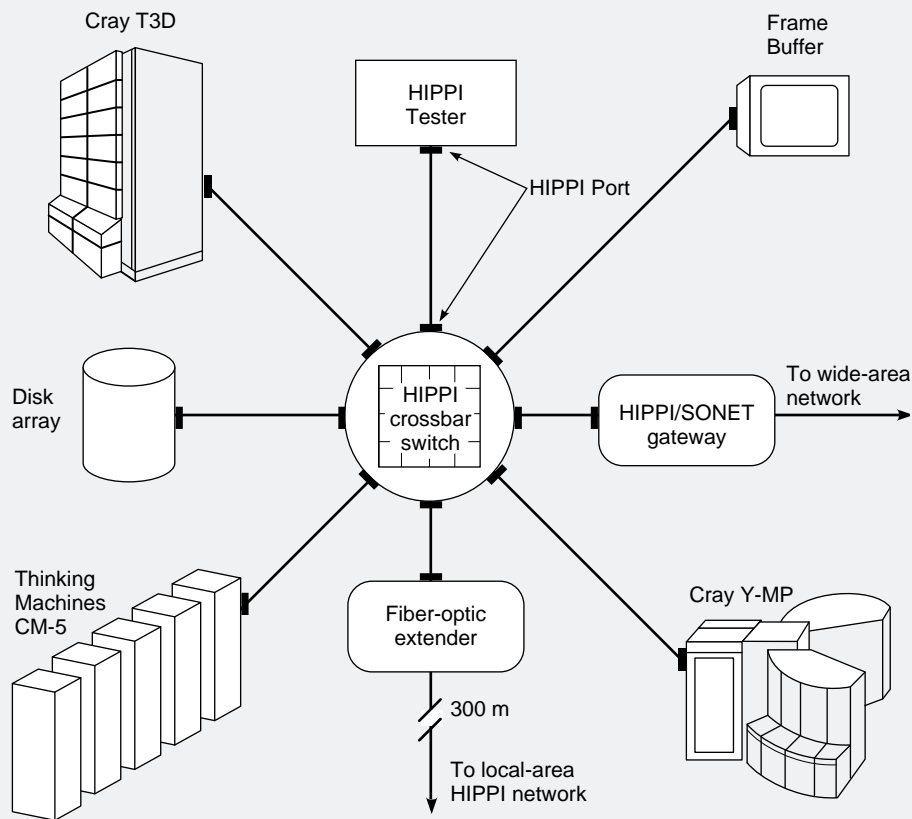


Figure 1. A Representative HIPPI Network

The figure shows a HIPPI crossbar switch connecting supercomputers, a high-performance disk array for file storage, a frame buffer, and a HIPPI tester (to monitor performance of the network). A fiber-optic extension connects this small network to another HIPPI switch, and a HIPPI/Sonet gateway connects the local HIPPI network to a wide-area SONET network. HIPPI cables are sets of 50 twisted-pair copper lines (for 800 million bits per second) or 100 lines (for 1.6 billion bits per second). The data transmission is parallel; that is, many bits are sent simultaneously, one bit per twisted-pair line. Hence, HIPPI cables are similar to the buses, or ribbons of wires, connecting parallel ports on personal computers. HIPPI ports contain the circuitry that organizes and interprets the parallel transmissions, sending 32 bits of data on the 32 data lines and other necessary signals on the other lines. An 800-million-bit-per-second HIPPI port uses a 25-megahertz clock to synchronize transmissions, so a 32-bit word is sent every 40 nanoseconds. If a device cannot supply a continuous stream of data, HIPPI allows idle time between groups of words.

dards based on new technologies are emerging, for example, Asynchronous Transfer Mode (ATM), but such technologies are not yet readily available.

HIPPI Components

The physical elements of the HIPPI system are ports for the devices that are connected to the network, cables for the network links, and a switch to connect the devices as desired. Figure 1 shows a small HIPPI network. (The HIPPI

Tester, the Frame Buffer, and the disk array are described below.) HIPPI was designed to be a “machine-room” network; the standards specify copper cables extending 25 meters or less. Vendors are now able to supply fiber-optic cables (governed by an implementors’ agreement, not a HIPPI standard) that can extend HIPPI links to a maximum distance of 10 kilometers so that HIPPI networks can cover wider areas.

The crossbar switch used in HIPPI networks is shown in more detail in Figure 2. The switch has multiple in-

puts and outputs and a separate connection path between every input and every output so that many simultaneous connections can be made. The original crossbar switches had sets of vertical and horizontal conducting strips as inputs and outputs—hence the term “crossbar.”

As the standards were being developed, Laboratory engineers were also developing equipment to help test and demonstrate HIPPI. The HIPPI Tester, for example, was developed to help ensure compatibility as well as to measure the performance of HIPPI ports developed by various computer manufacturers. The Tester, a briefcase-sized unit containing a lap-top computer and a special version of a HIPPI port, receives test signals from computers and checks to see if they were transmitted correctly; it also generates and transmits signals so that the received versions can be checked. Since the Tester is portable, Laboratory personnel were able to travel around the country to test the HIPPI ports being developed for various supercomputers. By 1990 the technology for the HIPPI Tester had been transferred to private industry. The Tester is easy to use, and now that it is manufactured in quantity, manufacturers and system designers can test their equipment themselves.

Another device developed at the Laboratory was the HIPPI Frame Buffer, a device for the high-speed display of computer graphics—which was the original motive for the development of HIPPI. A frame buffer consists of a large memory split into two buffers that each store a complete frame of display data. One buffer receives new data from a computer while the other buffer sends data to the display. When the “screen” buffer is empty, the roles of the two buffers are reversed. As long as the frame buffer contains data, the movie is not interrupted. The HIPPI

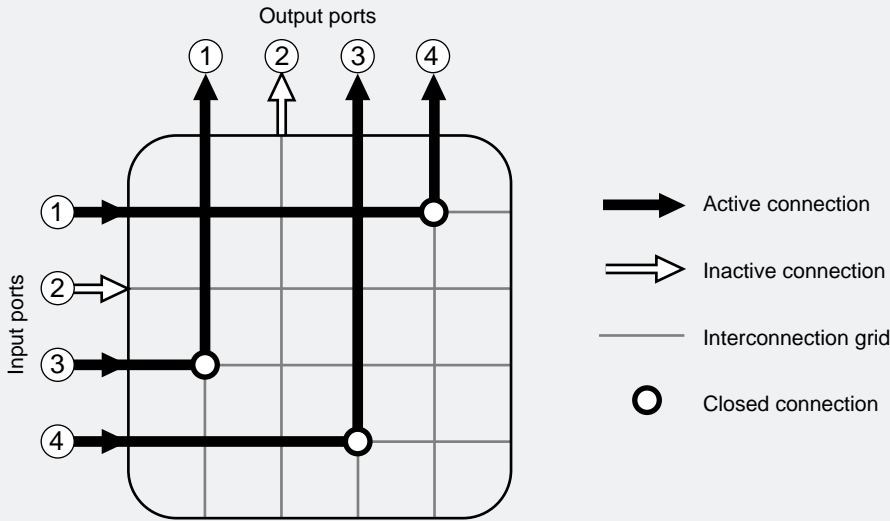


Figure 2. HIPPI Crossbar Switch

The figure illustrates a 4 x 4 crossbar switch, that is, one with four inputs and four outputs. There is a separate connection path between each input and each output, so the switch can connect any member of the network to any other as long as it makes no more than four simultaneous connections. (Similarly, a telephone switchboard, which is also a crossbar switch, can connect any telephone on the system to any other as long as the number of calls is not too great.) Three active one-way connections are shown: 1→4, 3→1, and 4→3. A typical HIPPI switch can connect up to 32 members in a HIPPI network (32 inputs and 32 outputs); several switches can be connected in a chain to make a larger network. Since one HIPPI connection can transmit data at 800 million bits per second, 32 simultaneous connections can transfer data at a maximum rate of 25.6 billion bits per second.

Frame Buffer was designed to be connected both to a supercomputer and to a high-resolution color monitor. The technology for this system was transferred to private industry in 1992.

As HIPPI networks were being tested, researchers realized that the bottlenecks in the networks were the supercomputers, not the HIPPI links. For example, a 30-second HIPPI frame-buffer movie requires about 24 billion bits of data. On the one hand, a typical Cray supercomputer could send data fast enough, but it had a memory capacity of only a few billion bits—less than what even a short movie requires. On the other hand, the new massively parallel computers had memory capacity several times the Cray capacity, but were

inefficient at protocol processing—packaging data for transmission and checking the data for errors. For example, when the massively parallel CM-2 was doing the protocol processing, it could supply data to a HIPPI port at only about 80 million bits per second or less.

In the last several years two new HIPPI components, the high-performance disk system (HPDS) and the crossbar interface (CBI), have been developed to solve these problems. An HPDS is an array of high-speed disks connected directly to a HIPPI port and controlled by a workstation. The array can hold more data and send data faster than the memory of a supercomputer—a typical disk array has a capacity of tens of billions of bits and can transfer

data at about 500 million bits per second. Data are transferred from a supercomputer or file storage to the disk array and then sent through the HIPPI network for analysis or display.

The CBI can boost the rate at which a massively parallel computer supplies data to a network. A CBI, which is a small, special-purpose computer, was originally developed to manage HIPPI networks. However, a CBI can also perform protocol processing. If data are sent out “raw” from the HIPPI port on the CM-2 and the protocol processing is done in a CBI, the effective transmission rate is about 400 million bits per second—a five-fold increase.

HIPPI and Wide-Area Networks

In 1991, as the official HIPPI standards were being completed, the National Science Foundation and the Advanced Research Projects Agency were beginning to build and evaluate wide-area gigabit networks. The Center for National Research Initiatives in Reston, Virginia, coordinated the establishment of five testbeds for gigabit-per-second communication and asked the Laboratory to join one of the projects, the Casa Gigabit Testbed. The other participants in the Casa testbed are the San Diego Supercomputer Center (SDSC) and the supercomputer centers at CalTech and NASA’s Jet Propulsion Laboratory (JPL). The goal of the project was to construct a HIPPI-based network connecting the four supercomputer centers and to use the network to investigate the applicability of distributed computing—several computers working together on the same calculation—to large computational problems. The problems designated were global climate modeling, seismic modeling, and chemical-reaction dynamics.

When the project started, each of the supercomputer centers had a local HIPPI network. To send HIPPI data over distances of hundreds or thousands of miles, the Casa network used another technology, the Synchronous Optical Network, or SONET, which the telephone companies had developed and were installing across the country as the next generation of telecommunication transmission. SONET is a network of fiber-optic transmission lines designed to transmit data for long distances at various rates of up to several billion bits per second. Engineers at the Laboratory developed a HIPPI/SONET gateway that moves data over the SONET network at the HIPPI rate of 800 million bits per second. The links between CalTech, JPL, and the SDSC were in place by August 1993; the Los Alamos link was brought up in June 1994 and connects a HIPPI switch at the Laboratory Data Communications Center at Los Alamos with a similar switch at the San Diego Computer Center. The project has now turned to distributed-computing research.

Current Research and Prospects

Laboratory researchers are now developing and evaluating new methods for gigabit-per-second transmission, developing links from HIPPI to other networks, and improving the performance of HIPPI networks. We are testing technology for a transmission standard called ATM, or Asynchronous Transfer Mode, which is under development and has the potential of reaching speeds of several gigabits per second (though the speed of most ATM transmissions in present testbeds is 155 million bits per second or less). When ATM transmissions can be made at speeds comparable to those of HIPPI, ATM will have a

great advantage since HIPPI can transmit only computer data, but ATM fully supports voice and video as well as data traffic. Moreover, ATM transmissions can be sent on SONET lines. Therefore ATM is being viewed as the network technology for the future national information highway. We are planning to develop a HIPPI/ATM gateway that will allow us to link the current HIPPI network to any future wide-area ATM network.

Another new technology uses optical transmission techniques with no electronic switching; optical switches direct light from a source to a destination. The Laboratory is helping to develop an optical method called wave-division multiplexing, which can send thousands of messages simultaneously on a single fiber-optic cable. Each message has a carrier frequency and is encoded as modulations of the light at that frequency; this method is similar to radio or television transmission. Current research on such networks is directed toward increasing the transmission speed to gigabit-per-second rates and increasing the speed of switching.

At present, though, HIPPI is the only well-developed technology for gigabit-per-second networks, and as an ANSI standard it will continue to be useful for at least the next five or ten years. When the use of fiber-optic networks becomes widespread, HIPPI will probably serve as a local-network backbone that connects to long-distance links for special applications. The Laboratory has had a prominent and crucial role in the development of HIPPI and continues to refine its capabilities. We are also helping to develop new methods of data transmission for the coming tenfold increase in network speeds. □

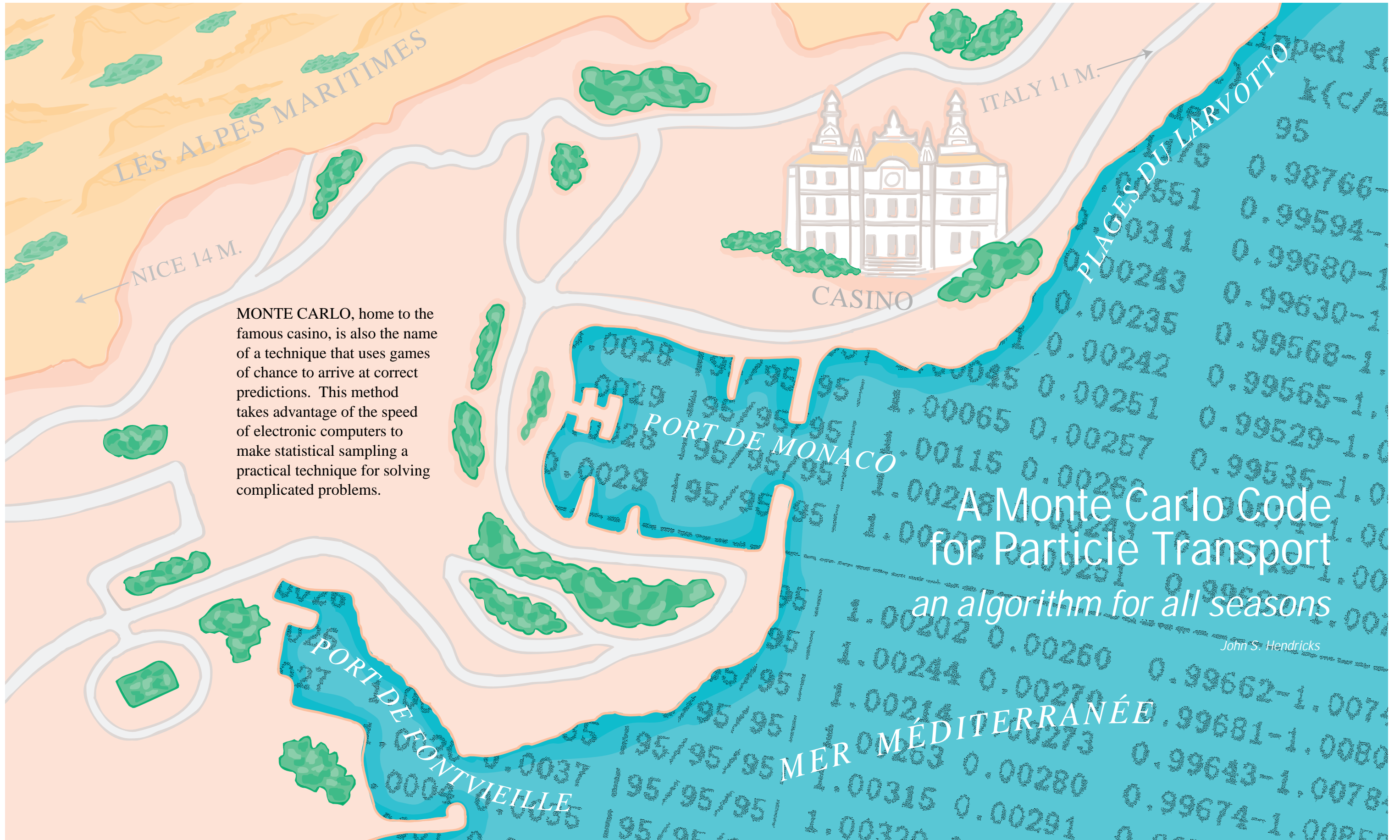
Further Reading

Don E. Tolmie and John Renwick. 1993. HIPPI: Simplicity yields success. *IEEE Network* 7: 28–32.

D. E. Tolmie. 1994. High performance parallel interface (HIPPI). In *High Performance Networks: Technology and Protocols*, edited by Ahmed N. Tantawy. Kluwer Academic Publishers.

Stephen C. Tenbrink is a technical staff member in the Network Engineering Group of the Laboratory's Computing, Information, and Communications Division. Since 1980 his focus has been on the development of high-performance computer networks. Tenbrink was the section leader for the group of engineers that developed the HIPPI-based systems at the Laboratory. Tenbrink received his B.S. from the University of Denver in 1970 and his M.S. in electrical engineering from the University of California, Los Angeles, in 1977.

Donald E. Tolmie has been involved in the networking of supercomputers since 1972. He was the chairman of the American National Standards Institute Task Group X3T9.3, which was responsible for HIPPI and other high-performance-transfer standards. Tolmie is a technical staff member in the Network Engineering Group of the Laboratory's Computing, Information, and Communications Division. He received his B.S. from New Mexico State University in 1959 and his M.S. in electrical engineering from the University of California, Berkeley, in 1961.



MONTE CARLO, home to the famous casino, is also the name of a technique that uses games of chance to arrive at correct predictions. This method takes advantage of the speed of electronic computers to make statistical sampling a practical technique for solving complicated problems.

A Monte Carlo Code for Particle Transport an algorithm for all seasons

John S. Hendricks

The Monte Carlo method was first applied on the MANIAC computer at the Laboratory to predict the rate of neutron chain reactions in fission devices. Now, half a century later, its recent incarnation as the MCNP code has become a significant technology-transfer success.

Fast computers and sophisticated computational techniques have launched a revolution in scientific research. No longer does that endeavor depend only on physical experimentation as a basis for developing and refining theory. Instead, scientific theories can now be based on numerical experiments, which, compared with physical experiments, not only are less expensive, considerably safer, and more flexible but also provide more information, a better understanding of physical phenomena, and access to a wider range of experimental conditions. In addition, analysis of the results of numerical experiments can guide the selection and design of the physical experiments best suited to validating theories. Thus numerical experimentation (also referred to as numerical modeling or numerical simulation) has added a new dimension to scientific research.

Two general approaches to numerical modeling are available—deterministic methods and the Monte Carlo method. Deterministic methods involve solution of an integral or a differential equation that describes the dependence on spatial coordinates or time of some behavioral characteristic of the system in question. The equation is cast in an approximate form that permits calculation of the incremental change in the characteristic caused by an incremental change in the variable(s). The value of the characteristic itself is then calculated at each of successive points on a spatial or temporal grid. The accuracy of deterministic methods is limited by how well the equation approximates physical reality and by the practical necessity of making the spatial or temporal difference between grid points finite rather than infinitesimal. Well-known deterministic methods include the finite-difference and finite-element methods.

The Monte Carlo method involves calculating the average or probable be-

havior of a system by observing the outcomes of a large number of trials at a game of chance that simulates the physical events responsible for the behavior. Each trial of the game of chance is played out on a computer according to the values of a sequence of random numbers. For that reason a Monte Carlo calculation has been defined in general as one that makes explicit use of random numbers. The Monte Carlo method is eminently suited to the study of stochastic processes, particularly the process called radiation transport—the motion of radiation, such as photons and neutrons, through matter. Another well-known use of the Monte Carlo method is the Metropolis technique for finding the equilibrium energy, at a given temperature, of a system of many interacting particles. The method also has a more strictly mathematical application, namely, estimating the value of complicated, many-dimensional integrals.

The principle behind the Monte Carlo method—statistical sampling—dates back to the late eighteenth century, but it was seldom applied because of the labor and time required. However, the mathematician Stanislaw Ulam, after returning to the Los Alamos laboratory in mid 1946, realized that the electronic computer, which had only recently become a reality, could turn statistical sampling into a practical tool. Ulam discussed his idea with the mathematician John von Neumann, a consultant to Los Alamos, who proceeded to outline a computerized statistical approach to a problem of immense interest to designers of nuclear weapons—neutron diffusion and multiplication (by fission) in an assembly containing a fissile material. Von Neumann sent his outline to the head of the Los Alamos Theoretical Division, and the idea was pursued enthusiastically. Among the Laboratory scientists who pioneered development of

the Monte Carlo method at Los Alamos, Nicholas C. Metropolis was the one who gave it its entirely appropriate and slightly racy name. Continuous effort at Los Alamos National Laboratory since those early times has culminated today in the computer code called MCNP (for Monte Carlo N-particle), perhaps the world's most highly regarded Monte Carlo radiation-transport code.

This article is of necessity too short to do justice to the Monte Carlo method. It cannot properly acknowledge the scientists who have devoted their careers to developing it or those who have successfully applied it in a variety of fields. All that will be attempted here is to describe the method, showcase a few of the many applications of MCNP, and to explain what is involved in developing and maintaining a modern Monte Carlo radiation-transport code such as MCNP.

The Monte Carlo Method

Portraying the essence of the Monte Carlo method is perhaps best accomplished by focusing on its application to radiation transport. For concreteness, let us focus in particular on the problem of estimating the probability that a neutron emanating from some source passes through some radiation shield. For simplicity assume that the source is an isotropic point source, that it emits monoenergetic neutrons, and that it is located at the center of the shield, which is a relatively thick spherical shell made up of matter containing only one isotopic species. The physics of the problem is well known: Each neutron emitted by the source follows a trajectory within the shield that consists of a succession of straight-line paths whose lengths and directions appear to be random relative to each other (Figure 1). That "random walk" is the

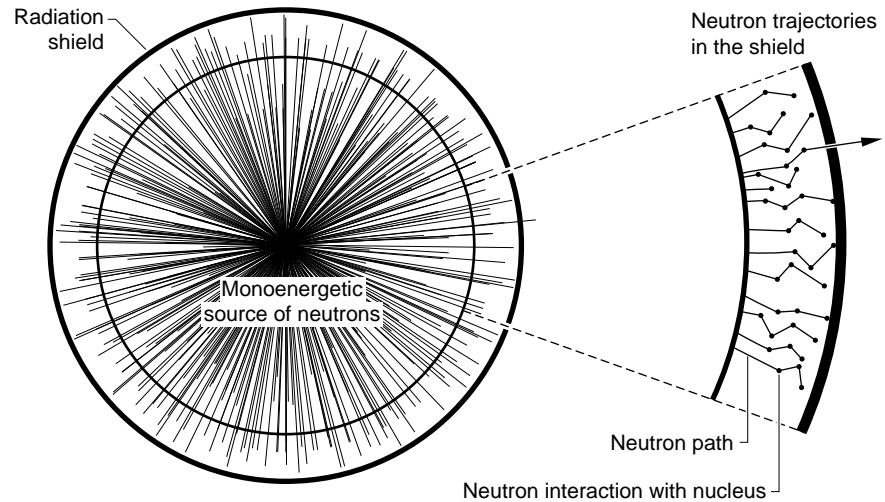


Figure 1. A Simple Monte Carlo transport Problem

In the problem discussed in the main text, a source of mono-energetic neutrons is at the center of a thin, spherical radiation shield. The object is to determine the probability that any given neutron emanating from the source will pass through the shield. As shown above, after entering the shield wall, each neutron will follow a succession of straight-line paths whose lengths and directions are in many respects random relative to each other. In other words, the neutron's trajectory resembles a "random walk". Both changes in direction and terminations of a given trajectory result from the neutron's interaction with the nuclei in the shield. The possible interactions with the nuclei in the shield are: elastic scattering, which changes the neutron's direction of motion but not its energy; inelastic scattering, which changes both the direction of motion and the energy of the neutron; absorption, which terminates the neutron's trajectory as the neutron is absorbed into the nucleus; and fission, which produces additional neutrons but occurs only if the shield contains certain isotopes. The length of the path between one interaction and the next as well as the outcome of each interaction are described by probability distributions that have been determined experimentally. The Monte Carlo method is used to construct a large set of possible trajectories of a neutron as it travels through the shield; the experimental probability distributions are sampled during the construction of each trajectory. The neutron's probability of escape is then determined based on the outcomes in that set of trajectories. The figure shows some possible neutron histories in the enlarged view of the shield wall. Most of them scatter a number of times before being absorbed and one escapes through the shield.

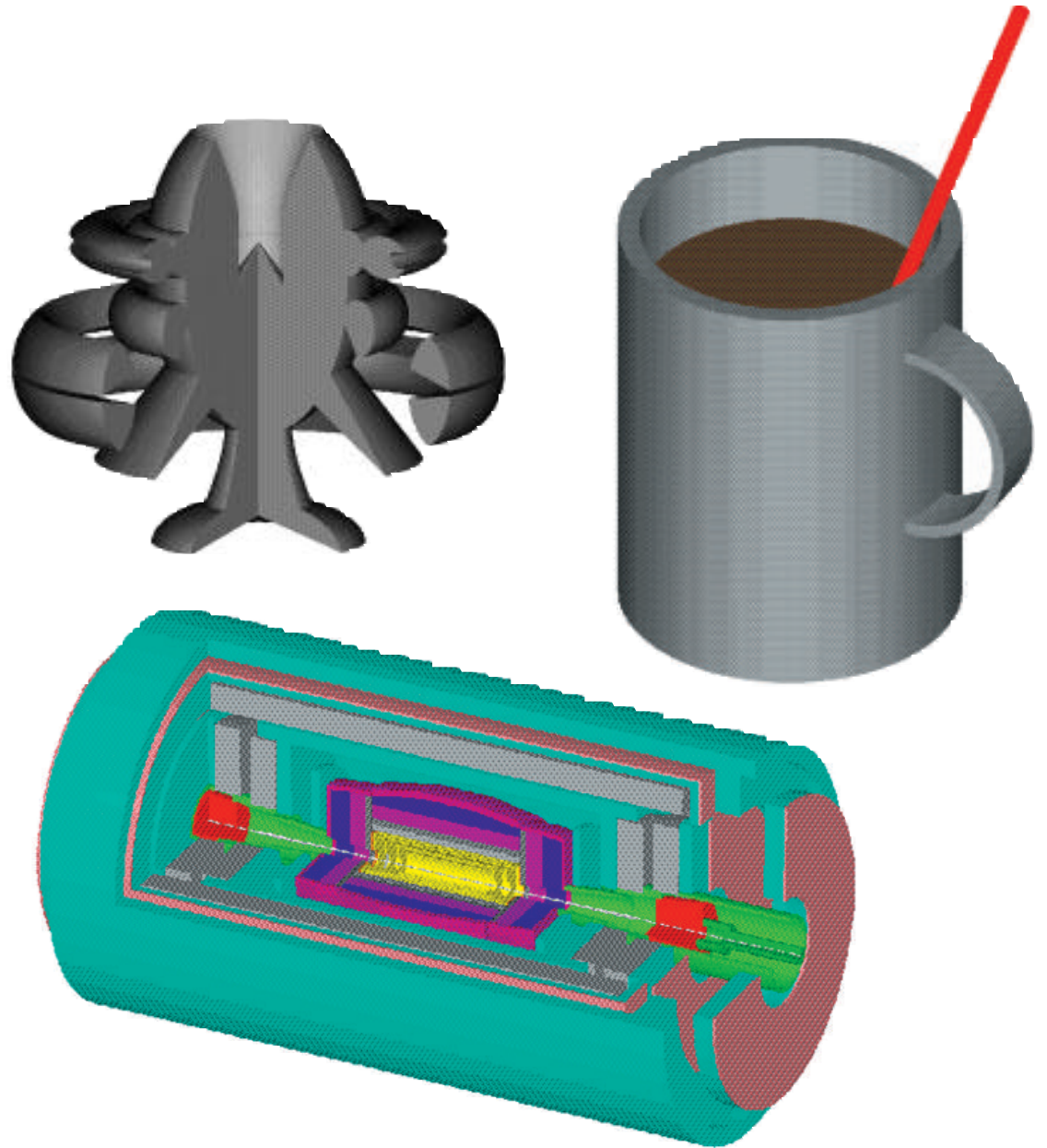


Figure 2. Geometry Modeling in MCNP

Shown here are examples of the geometry-modeling capability that can be accessed for MCNP calculations. The geometries were created and visualized with the graphics software known as Sabrina, which was developed by James T. West and Kenneth Van Riper. Each example is depicted in a different visualization format. Unlike the other examples, the example in color, a model of the SDC detector for the Superconducting Super Collider, was constructed not simply to demonstrate the capabilities of Sabrina but for use in an MCNP simulation. The detector, shown here in cutaway view, contains dozens of components and is very large (the outer cylindrical surface shown has a diameter of nearly 22 meters). The accelerator was designed to produce two counter-propagating beams of extremely energetic (20-TeV) protons that travel along the axis of the detector and collide at its midpoint. The detector model was used in MCNP simulations to track the products of the collisions and to calculate the levels of the radiation induced in the detector components by the collision products. The radiation levels are necessary to assess component damage and background signals in the detector.

result of interactions of the neutron with nuclei within the shield. The possible types of interactions of a neutron with a nucleus include elastic scattering, inelastic scattering, absorption, and fission. For simplicity, assume that the only interactions that occur are elastic scattering, which changes the direction but not the speed (and hence energy) of a neutron, and absorption, a nuclear reaction that swallows up, or “kills,” a neutron. Whether an interaction results in a neutron’s being absorbed or scattered can be predicted only probabilistically, as can the scattering angle if the neutron is scattered. The probabilities of a neutron’s being absorbed by various isotopes have been measured, and so have the probabilities of its being scattered through various angles, all as functions of neutron energy. Those probabilities, or cross sections, for the shield nuclei are necessary input to solving the problem at hand. Also needed is the probability density function for the distance a neutron travels in the shield without undergoing an interaction with a nucleus (in other words, the probability density function for the lengths of the straight-line paths composing the neutron’s trajectory). It is known that the probability density function for the “free-path” length in any material decreases exponentially. In particular, the probability density that a neutron will travel a distance x before undergoing an interaction is given by $\rho\sigma e^{-\rho\sigma x}dx$, where ρ is the density of nuclei and σ is the total cross section (here the sum of the scattering cross section integrated over scattering angle and the absorption cross section).

Application of the Monte Carlo method to the problem above involves using a sequence of numbers uniformly distributed on the interval (0, 1) to construct a hypothetical (but realistic) history for each of many neutrons as it travels through the shield. (To say that

a sequence of numbers is uniformly distributed on (0, 1) means that any number between 0 and 1 has an equal probability of occurring in the sequence. Such numbers, when generated by a computer, are called pseudorandom numbers.) The ratio of the number of neutrons that escape from the shield to the number of neutrons whose histories have been constructed is an estimate of the answer to the problem, an estimate whose statistical accuracy increases as the number of neutron histories increases. Details of the process can be illustrated by following the construction of a single neutron history.

Constructing the first step of a neutron history involves deciding on a value for its first free-path length x_1 . As pointed out above, the sequence of pseudorandom numbers generated by the computer is uniformly distributed on (0, 1), whereas the free-path lengths are distributed according to $e^{-\rho\sigma x}$ on (0, ∞). How can the sequence of uniformly distributed numbers ξ_i be used to produce a sequence of numbers x_i whose distribution mirrors the experimentally observed distribution of free-path lengths? It can be shown that the transformation $x_i = -(1/\rho\sigma)\ln(1 - \xi_i)$ yields a sequence of numbers that have the desired inverse exponential distribution. So the first free-path length is obtained by setting $x_1 = -(1/\rho\sigma)\ln(1 - \xi_1)$. The second step in the neutron history involves deciding whether the neutron’s first interaction with a nucleus scatters or kills the neutron. Suppose it is known from the cross sections for the shield nuclei that scattering is nine times more likely than absorption. The interval (0, 1) is then divided into two intervals, (0, 0.1) and [0.1, 1). Assume that x_2 , the second pseudorandom number generated by the computer, is 0.2. Because 0.2 lies within the larger subinterval, the neutron is scattered rather than absorbed. The third step in

the neutron history involves deciding through what angle it is scattered. Again some transformation must be performed on the third pseudorandom number, a transformation that changes the uniform distribution of the ξ_i into a distribution that mirrors the observed distribution of scattering angles (the scattering cross section as a function of scattering angle). Further steps in the history are generated until the neutron is absorbed or until the radial distance it has traveled within the shield exceeds the thickness of the shield. The histories of many more neutrons are generated in the same manner.

Assume that N neutron histories are generated and that n of the histories terminate in escape of the neutron from the shield. To calculate an estimate for the probability that any single neutron escapes, assign a “score” s_i to each neutron as follows: $s_i = 0$ if the neutron is absorbed within the shield, and $s_i = 1$ if the neutron escapes. Then the estimated probability of escape is given by the mean score \bar{s} , that is, by $(1/N)\sum s_i = n/N$. The relative error (relative statistical uncertainty) in that probability estimate is related to the so-called variance of the s_i , $\text{Var}(s_i)$, which can be approximated, when N is large, by the difference between the mean of the squares of the scores and the square of the mean score:

$$\begin{aligned}\text{Var}(s_i) &\approx \frac{1}{N} \sum_i s_i^2 - \left(\frac{1}{N} \sum_i s_i \right)^2 \\ &= \frac{n(N-n)}{N^2}\end{aligned}$$

The relative error in the probability estimate is then given by

$$\frac{\sqrt{\text{Var}(s_i)/N}}{\bar{s}} = \sqrt{(N-n)/Nn}$$

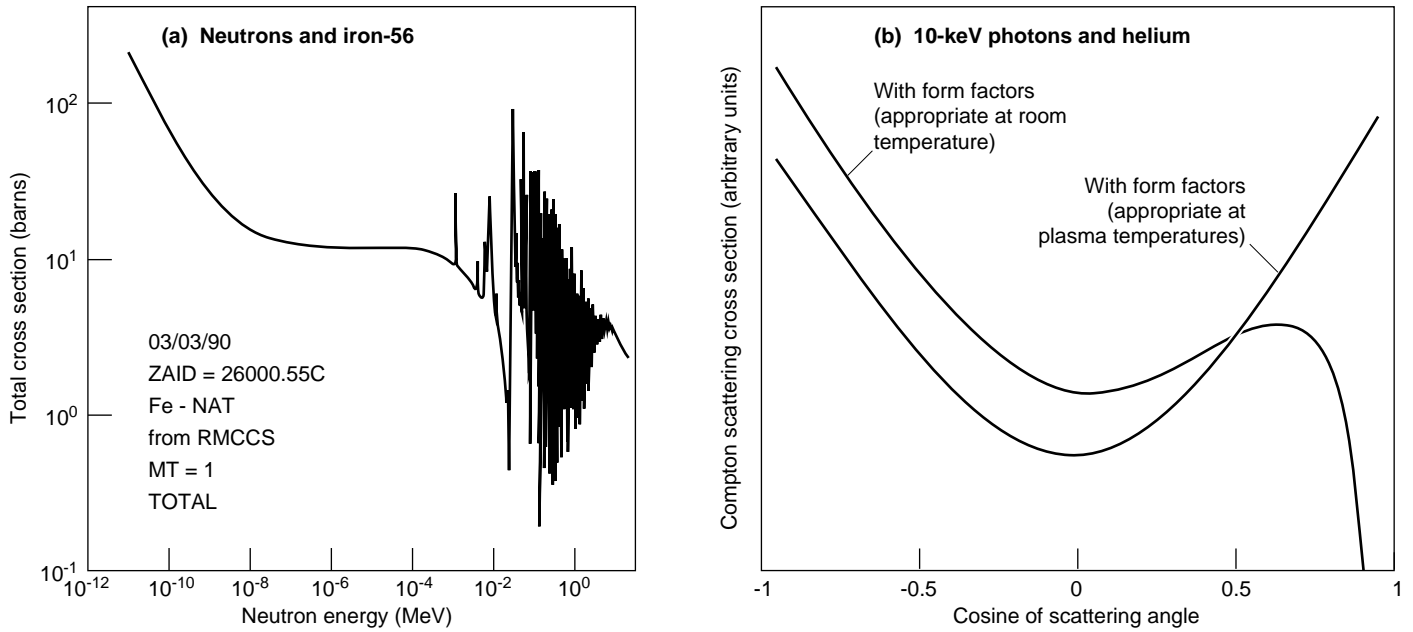
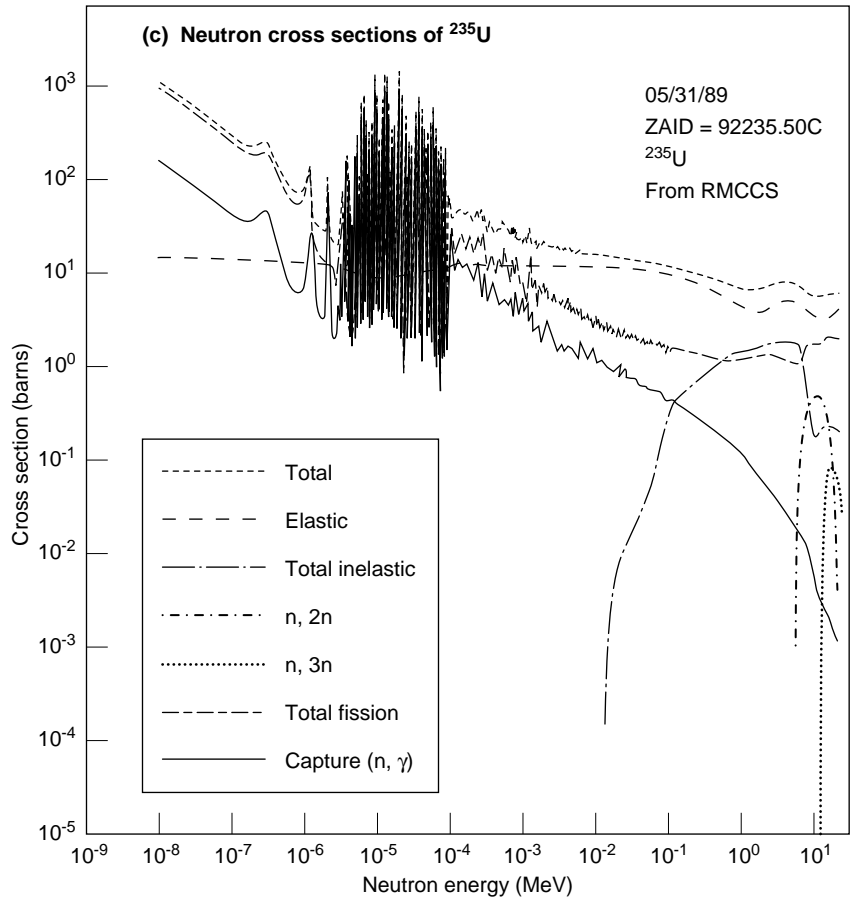


Figure 3. A Few Selections from the MCNP Data Library
 The MCNP data library contains a vast amount of information about the interaction of radiation with matter. Shown here are three items included in that library: (a) the continuous-energy representation of the total cross section, as a function of neutron energy, for the interaction of neutrons with the most abundant stable isotope of iron, iron-56; (b) the “differential” cross section (cross section as a function of scattering angle), with and without the form factors that account for electron binding, for Compton scattering of 10-keV photons off helium; (c) the cross sections, as functions of neutron energy, for various interactions of neutrons with uranium-235. Total, elastic, and inelastic cross sections are shown, as well as cross sections for capture of the incoming neutron accompanied by the emission of a photon and for reactions in which one incoming neutron results in two or three outgoing neutrons, (n, 2n) and (n, 3n).



For example, if $N = 100$ and $n = 47$, the probability of escape is estimated as 0.47 and the relative error is a little less than 11 percent.

Now is a good time to point out a major difference between a Monte Carlo calculation and one based on a deterministic method. If the calculation outlined above is repeated, the second calculation is highly unlikely to show that exactly 47 of the 100 histories result in escape. In other words, repetitions of a Monte Carlo calculation yield only approximately the same answer, whereas repetitions of a calculation based on a deterministic method yield exactly the same answer. That difference is due to the element of chance inherent in a Monte Carlo calculation. Thus deterministic methods provide more exact solutions of approximate models, whereas Monte Carlo methods provide approximate solutions of more exact models.

The simplifying assumptions invoked in the above example of a Monte Carlo radiation-transport calculation do not of course hold in general. The radiation may consist of particles other than neutrons (or of more than one type of particle), and other types of interactions may be involved (neutron-induced fission of fissile nuclei, for example). The radiation may not be monoenergetic, and it may emanate from a source that is neither point-like nor isotropic. The material through which the radiation travels may be nonuniform in composition and intricate in geometry. All those additional complexities can be handled provided the necessary input data are available.

Perhaps the greatest advantage of using the Monte Carlo method to simulate radiation transport is its ability to handle complicated geometries. That ability rests on the fact that, even though the geometry in question may be complicated in its entirety, only the

geometry in the vicinity of the particle for which a random walk is being constructed need be considered at any point in the construction. A given geometry can be modeled in its entirety in two ways: as a “combinatorial” object or as a “surface-sense” object. As its name implies, a combinatorial object is constructed by combining relatively simple geometric entities, such as rectangular parallelepipeds, ellipsoids, cylinders, cones, and so on. Combinatorial objects are easy to construct but are less general than the surface-sense objects provided by MCNP. A surface-sense object is constructed by combining bounding surfaces, each of which is assigned one of two sense values to indicate on which side of the bounding surface the object lies. Any combination of linear, quadratic, or toroidal surfaces in any orientation or even skew can be accessed by users of MCNP. Figure 2 shows examples of the intricate geometries that can be constructed for MCNP calculations.

It is worth noting that a Monte Carlo radiation-transport code is more than a list of instructions for executing a certain set of arithmetic and logic operations. It is also a repository of experimental data about and theoretical understanding of radiation transport accumulated over the years. That knowledge is accessed unwittingly by users whose expertise may lie in areas far removed from radiation transport. The data are not built into modern Monte Carlo radiation transport codes but rather are available as separate data libraries. That stratagem permits the data libraries to be upgraded independently of the operational portion of the code. Many older Monte Carlo codes include “multigroup” data, which are averaged over ranges of energy or of some other parameter. That practice saves considerable data-storage space but introduces approximations into the

calculations. MCNP and other modern Monte Carlo radiation-transport codes offer libraries of unaveraged, or “continuous-energy” data. Shown in Figure 3 are a few specific examples of the data contained in the MCNP data library.

The example of neutron transport through a spherical-shell shield shows that a Monte Carlo radiation-transport calculation is similar to an opinion poll, such as a Gallup poll. Both use the technique of statistical sampling to obtain a probabilistic answer to some question. Just as a Gallup poll attempts to predict the outcome of, say, an election by questioning only a small portion of the voting population, so also does a Monte Carlo radiation-transport calculation attempt to simulate the outcome of the interaction of 10^{15} to 10^{25} neutrons with some material by constructing histories for 10^5 to 10^8 neutrons. And just as the voters questioned during a Gallup poll are not chosen randomly but are carefully selected to mirror demographic characteristics of the entire voting population, so also are the neutron histories not truly random walks but histories that mirror the observed phenomenology of neutron interactions with matter. The analogy between a Gallup poll and a Monte Carlo radiation-transport calculation will be continued in the following discussion of variance reduction.

Variance Reduction

As pointed out above, the result of a Monte Carlo calculation has associated with it a statistical uncertainty. How can that uncertainty be reduced and the result thereby be made more accurate? One obvious way to do so is to increase the number of neutron histories generated (or voters questioned). But that “brute-force” approach is costly in

terms of computer time (or pollster time). Fortunately, more sophisticated techniques are available to achieve a lower uncertainty without increasing the number of histories (voters questioned) or to achieve the same uncertainty from a smaller number of histories. Four types of such “variance-reduction” techniques are available: truncation, population control, probability modification, and pseudodeterministic methods.

Truncation involves ignoring aspects of the problem that are irrelevant or inconsequential. For example, the source-and-shield assembly described above may include structural elements that position the source at the center of the shield. Because the nuclei in the structural elements, like the nuclei in the shield, interact with the neutrons, the structural elements should be included in the simulation. Suppose, however, that the structural elements are very fine rods and hence are considerably less massive than the shield itself. Then truncating the problem by ignoring the existence of the structural elements would have little effect on the results. An example of truncation in a Gallup poll is to not include among the sampled population those who live abroad and yet are eligible to vote because such persons are unlikely to affect the outcome of an election.

Population control involves sampling more important portions of the sampled population more often or less important portions less often. For example, suppose the neutrons that escape from the left half of the spherical-shell shield are of greater interest (say, because someone’s office is located there, whereas a little-used stairwell is located to the right) and that the left half of the spherical shell is composed of a material more effective at absorbing neutrons. Each neutron that has a possibility of reaching the region of greater interest (any neutron that is emitted toward the

left) is “split” into m neutrons ($m > 1$) and assigned a “weight” of $1/m$. The scores of the histories of the split neutrons are multiplied by their weight so that the splitting stratagem does not alter the physical situation but does allow the sampling of more of the more important neutrons. The corresponding technique for sampling fewer of the less important neutrons is referred to as Russian roulette. Applied to the same example, Russian roulette involves specifying that the neutrons emitted to the right have a probability of $(1 - 1/m)$ of being terminated immediately upon entering the shield. A neutron whose history begins with immediate death is of course tracked no further. Those neutrons that do not suffer immediate death, $(1/m)$ of the neutrons emitted to the right (in the limit of large N), are assigned a weight of m . Thus the simulation of the real physical situation is unaltered. An application of population control in a Gallup poll might be as follows. Suppose 40 percent of the voters are Democrats, 40 percent are Republicans, and 20 percent are independents. Then the outcome of the vote on an issue such that Democrats and Republicans are likely to vote the party line is determined primarily by the independent vote. Therefore a good polling strategy would be to question a sample consisting of 20 percent Democrats, 20 percent Republicans, and 60 percent independents (such a sample could be achieved by including every independent on a list of 300 registered voters but including each party member only if the roll of a die yielded a chosen one of the six possible outcomes) and weight the opinions of the 20 Democrats and the 20 Republicans by a factor of 2 and the opinions of the 60 independents by a factor of $1/3$. The discarding of 100 Democrats and 100 Republicans corresponds to their death by Russian roulette.

Probability modification involves sampling from a fictitious but convenient distribution rather than the true distribution and weighting the results accordingly. For example, instead of applying splitting and Russian roulette to the neutrons emitted to the left and right, respectively, by the isotropic neutron source, the spatially uniform neutron distribution is replaced, for the purpose of constructing histories, by a distribution such that more neutrons are emitted to the left. The “bias” that such a strategy would introduce into the result is removed by appropriately weighting the scores of the neutron histories. An example of probability modification in the Gallup-poll analogy might be the following. Suppose only 40 percent of the voters questioned by a pollster happened to be women. Instead of questioning sufficiently more women to bring the sexual distribution of the sampled population to a 50-50 distribution, the response of each woman could be weighted by a factor of $5/4$ and that of each man by $5/6$.

Pseudodeterministic methods are among the most complicated variance-reduction techniques. They involve replacing a portion (or portions) of the random walk by deterministic or expected-value results. Suppose, for example, that the spherical shell is surrounded by further shielding material with complex geometry. Instead of transporting each neutron via a random walk through the spherical shell to the more complex region of the shield, the neutron may simply be put at the interface between the two shield components and assigned a weight equal to the (presumably known) probability of its arriving there. Similarly, it is known that not all those eligible to vote are equally likely to carry out their civic duty. So the response of each person polled may be assigned a weight equal to the probability, based on factors such

as gender, race, age, and place of residence, that he or she will indeed vote. The difficulties encountered when using pseudodeterministic methods arise in assigning the probabilities.

The use of modern variance-reduction methods has allowed Monte Carlo calculations to be carried out many orders of magnitude faster and yet maintain the same statistical accuracy. In fact, many calculations that once would have required prohibitive amounts of computer time are now routine.

Applications of Monte Carlo Radiation-Transport Codes

The many applications of the Monte Carlo method in radiation transport reflect the pervasiveness of radiation in nature and in established and emerging technologies. A few of the applications of the Laboratory's MCNP code are discussed below.

Criticality safety. Preventing the inadvertent assembly of critical masses of fissile material is a grave concern at facilities that process fissile materials, such as the laboratories of the DOE's nuclear-weapons complex and facilities associated with the generation of nuclear power, including not only the power plants themselves but also waste-storage sites and fuel-fabrication plants. An MCNP simulation is a safe, reliable way to assess, for example, whether a given amount of fissile material assembled in a given geometry constitutes a critical mass or whether a chemical reaction involving a fissile material or a change in physical state of a fissile material can lead to the assembly of a critical mass. Answering such questions by physical experimentation is so dangerous as to be out of the question in many instances.

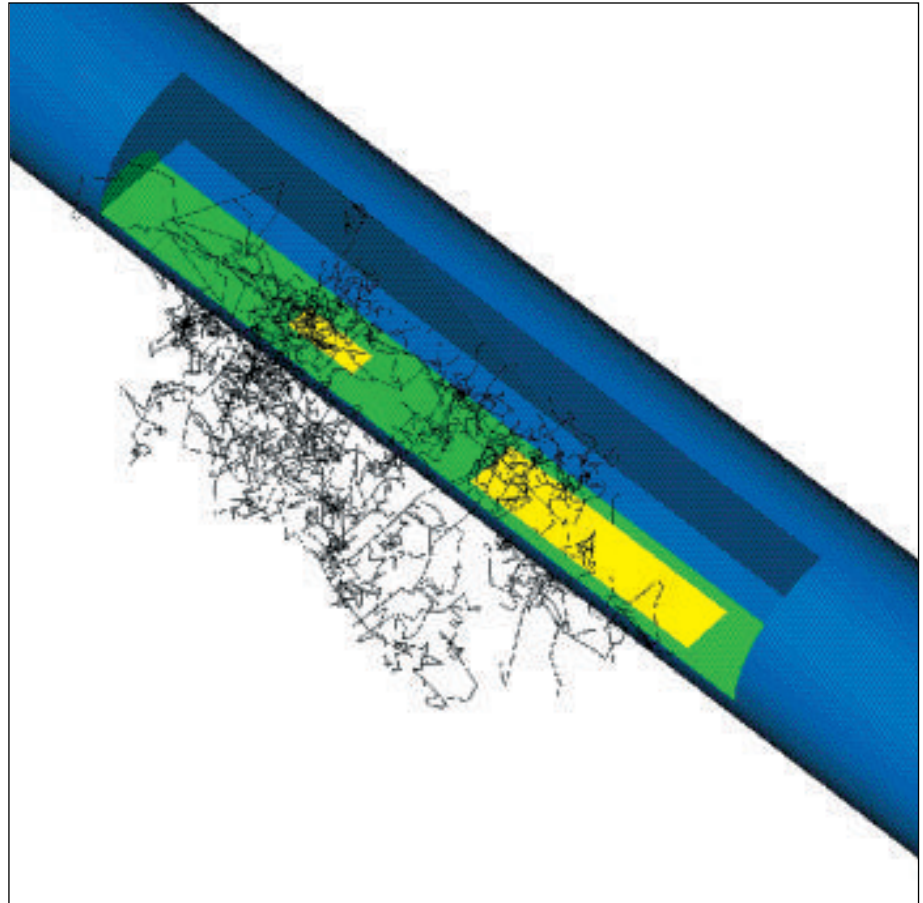


Figure 4. Nuclear Tool for Oil-Well Logging

This three-dimensional view of a generic nuclear tool for oil-well logging was generated with the graphics software Sabrina, which is linked to MCNP. The cutout displays a helium-3 neutron-detector assembly (yellow) encased in iron (green) and immersed in a water-filled borehole (blue), which is surrounded by a limestone formation (brown). Neutrons emanating from a source are scattered from the rock formation into the detector assembly. The red lines are neutron paths, as simulated by MCNP.

The MCNP code is used worldwide to predict the transport of radiation in a dazzling array of applications from oil-well logging and fusion research to the most advanced medical diagnostics and treatments. It is adapted to run on supercomputers, but even more important, it can be used by nonspecialists on ordinary personal computers. With over a thousand users at a hundred institutions around the world, the MCNP code is one of the most successful in the history of scientific computing.

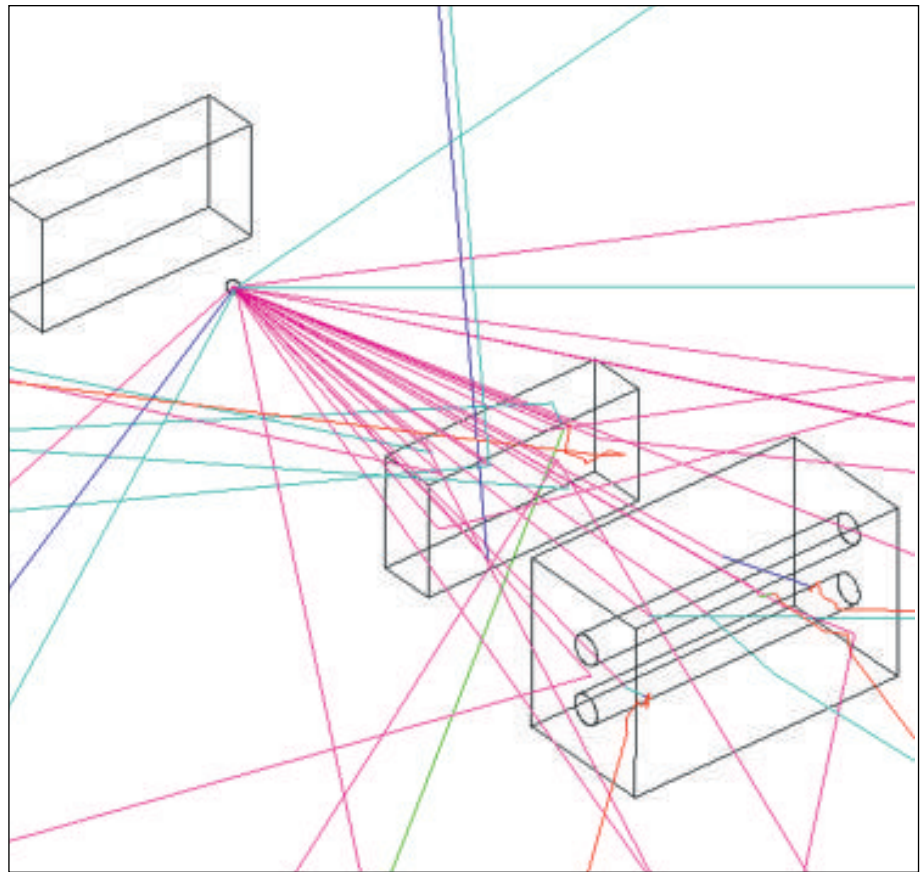


Figure 5. Example of a Particle-History Plot

Shown here are the MCNP-simulated histories of neutrons as they travel from a source, pass through a lead brick, and interact with helium-3 within two neutron detectors (thin cylinders). The neutron tracks are color-coded according to energy. The plot was generated with the graphics software Sabrina.

Oil-well logging. Surprising as it may be, radiation and hence MCNP play important roles in the search for oil. (Gone are the days when “finding oil” was equated with the gushing of oil from an exploratory hole.) The pattern of scattering of neutrons and gamma rays from rock formations can indicate the presence or absence of oil. MCNP can predict the scattering pattern characteristic of an oil-bearing rock formation, which is then compared with the scattering pattern obtained with a “logging” tool inserted into an exploratory bore hole drilled in the rock formation. As shown in Figure 4, the logging tool contains a source of neutrons and a detector assembly for observing the radiation scattered from the rock. Such a nuclear logging tool is particularly effective when used in conjunction with other logging tools that contain a source of radio waves or sound waves,

whose scattering provides additional information. MCNP is used by many companies around the world in the search for oil. The Gas Research Institute and Schlumberger-Doll Research, a major oil-service company, have contracts with the Laboratory for continued development of MCNP.

Nuclear energy. MCNP serves a variety of purposes in the hundreds of nuclear power plants around the world. It is used, for example, to design shielding and spent-fuel storage ponds and to estimate the radiation dose received by operating personnel. General Electric Nuclear Energy of San Jose, California, and other fuel-assembly manufacturers use MCNP to design assemblies so that the fuel is “burned” more efficiently. More efficient burning implies cheaper electricity and less nuclear waste.

Nuclear safeguards. The signing of the Treaty on the Nonproliferation of Nuclear Weapons in 1968 was one of the most important political accomplishments of this century. Those signatory nations that had not yet developed their own nuclear weapons agreed to forego such development in return for access to peaceful nuclear technology. A cornerstone of the treaty is inspection of nuclear facilities by the International Atomic Energy Agency to ensure that no "special" nuclear materials are stolen from the facilities. The inspectors use nondestructive techniques based on radiation detection to assay the uranium-235 or plutonium-239 content of, say, fuel assemblies, barrels of nuclear waste, and workers' lunch boxes. The nondestructive assay techniques are based on detecting either the radiation emitted by those isotopes themselves in the course of radioactive decay or the radiation emitted by the products of nuclear reactions induced in the isotopes by irradiation with neutrons or gamma rays. Many of the detectors used by the inspectors were designed with the help of MCNP.

Fusion research. For years scientists have been investigating the possibility of producing energy by nuclear fusion, the mechanism that powers our sun. Nuclear fusion offers two advantages over nuclear fission as an energy source: It does not produce hazardous fission products, and it consumes a naturally abundant fuel—deuterium, an isotope of hydrogen that can be extracted from sea water. The most advanced strategy for fusing deuterium nuclei involves containment and compression of a plasma of deuterium ions by a toroidal magnetic field. Since MCNP is one of the few Monte Carlo radiation-transport codes capable of handling the challenging fourth-order toroidal geom-

etry, it has long been the premier code for studying the transport of the neutrons and photons produced by the fusion reaction. Another use of MCNP in fusion research is studying ways to prevent damage to personnel and equipment by the fusion-produced radiation.

Medical technology. Few people realize that medicine is the third largest application of nuclear reactions, ranking behind only energy and defense. Among the nuclear medical technologies are boron neutron-capture therapy and positron-emission tomography. Boron neutron-capture therapy involves injecting a cancer patient with the stable isotope boron-10 and then irradiating the patient with neutrons. Because boron-10 collects preferentially in tumor cells and has an exceptionally high cross section for capturing (absorbing) neutrons, the therapy selectively kills tumor cells. MCNP is used to determine the neutron dose and energy spectrum that will kill the tumor and not the patient. Positron emission tomography is a nondestructive technique for observing metabolism in situ. It involves ingestion by the patient of water containing the radioactive isotope oxygen-15 rather than the usual, nonradioactive isotope oxygen-16. As the water is used by the body, a scanner detects the gamma rays resulting from interaction of the positrons emitted by the oxygen-15 with electrons in cells. For example, a PET scan may reveal that part of a heart is dead and that thus the heart requires replacement rather than repair. MCNP is used to properly interpret PET-scan images, which are blurred by the effects of electron and photon scattering. MCNP benefits many other medical technologies, including computer-assisted tomography, radiation therapies for cancer, and protection of medical personnel from nuclear radiation and x rays.

Space exploration. Among the harsh features of space are intense bursts of radiation. Simulating the effects of that radiation on equipment and personnel is a crucial preliminary to the exploration of space, particularly in light of its high cost. Calculations with MCNP and other codes have helped design shielding of minimum weight to protect astronauts from cosmic rays. In one case a "storm shelter" had been designed to protect astronauts from solar flares, but calculations with MCNP showed that although the proposed shielding would block most of the incident protons, so many gamma rays would be generated in the shield that the astronauts were safer outside the storm shelter than inside it! MCNP calculations have also shown that nuclear power will be essential to extended space exploration. The extra radiation astronauts would receive from a nuclear reactor is far less than the additional cosmic-ray radiation they would be exposed to during a slower, longer journey. Of course, MCNP plays a role also in designing nuclear-power systems for use in space and in assessing the survivability of sensitive electronic components and means to protect them from damage.

Applications at the Laboratory

Since MCNP was developed here to help carry out the Laboratory's primary mission, it is not surprising that the number of MCNP users here constitutes a large fraction of the total worldwide. Some of the applications at the Laboratory have already been mentioned: criticality-safety studies and the design of detectors for implementation of the Nonproliferation Treaty and for space exploration. The long list of other Laboratory uses of MCNP includes design of the neutron-producing target at the

Manuel Lujan, Jr. Neutron Scattering Center (LANSCE); design and evaluation of apparatus to carry out the proposed ATW (accelerator transmutation of waste) scheme for burning surplus plutonium and transmuting long-lived fission products into stable or short-lived isotopes by irradiation with accelerator-produced neutrons; design of the dual-axis radiographic hydrodynamic test (DARHT) facility for x-ray imaging of explosions; and evaluation of the radiation hazard posed by the plutonium-preparation line at the Laboratory's plutonium-processing facility.

Challenges of MCNP Development

The maturity of the Monte Carlo method and the high satisfaction level of MCNP users might imply that a large investment in future development is unnecessary. After having gone only a few years ago through the formidable process of certifying that version 4 satisfied rigid quality-assurance standards, why should we now be willing to go through the same process for version 4A? Or why should a user already happy with MCNP version 4 bother to obtain and implement version 4A? And finally why should the sponsors of MCNP continue to spend money on a code that is already "good enough"? The answers to those questions lie in the challenges presented by the shifting computer hardware and software environments, by the many users of MCNP, and by the demand for a code of increasing versatility and sophistication.

The MCNP of ten years ago would not run on today's computers because the architectures have changed. MCNP must constantly be adapted to new architectures and other new aspects of the computer world. But care must be taken not to squander limited development

funds on inappropriate adaptations. For example, we have deliberately avoided adapting MCNP to massively parallel computers that do not use UNIX, standard FORTRAN 77, and MIMD (multiple-instruction, multiple-data) architectures. If massively parallel machines are to become the computers of the future, they will have to accommodate to UNIX and FORTRAN 77, most likely in a MIMD architecture. For similar reasons we have also avoided adapting to systems with immature software and compilers. However, MCNP is usually among the first production physics codes to become available on any commercially viable, state-of-the-art computer architecture. For example, the latest version of MCNP (version 4A) takes full advantage of parallel processing on a cluster of workstations, an architecture that offers many times the computing performance of a single-processor Cray supercomputer.

Strange as it may seem, adapting MCNP to avant-garde architectures is currently of lesser interest than adapting it to the most primitive of computers, the IBM-PC and its clones, which use an operating system, MS-DOS, not designed for scientific computing. The performance of MCNP on such a machine, equipped with a larger memory and a faster processor (such as the 486 chip) and costing less than \$3000, approaches its performance on today's supercomputers. In fact, the performance-to-cost ratio is about 10,000 times better than that available on the supercomputers of just a few years ago!

Updating the graphics package included in MCNP also requires considerable effort because graphics packages are not as standardized as operating systems and languages. A major enhancement to the latest version of MCNP is the addition of color graphics and an X-windows-based graphics package. Additional links have also

been made to an auxiliary code, Sabrina, which allows geometries to be sketched (see Figure 2) and publication-quality plots of particle histories to be generated. Figure 5 shows an example of such a particle-history plot.

Of all the physics computer codes that have been developed over the years, MCNP is among the leaders in number of users—over a thousand at about a hundred institutions. The Laboratory is obligated, for scientific, ethical, and regulatory reasons, to make MCNP generally available. But meeting the challenge of providing users with a minimum level of support despite the lack of explicit funding for such support remains a dilemma.

Continued maintenance and development are also required to meet new criteria, which may be regulatory, technical, or even political. Higher standards of accountability, particularly in the area of health, safety, and environment, have generated an avalanche of government requirements concerning the quality-control standards imposed on computer codes, including physics codes. Most of the requirements are long overdue, and most have already been met by MCNP. All require voluminous paperwork and are an increasing part of the MCNP-maintenance effort. New technical criteria come about from advances in science and new applications. For example, as new libraries of data about the interactions of neutrons, photons, and electrons with matter become available, MCNP must be upgraded to accommodate different sampling techniques and data formats. New applications may require different code features, such as periodic boundaries and enhanced tally options. Modifications to MCNP are also required for linkage to related computer codes, such as the Sabrina graphics code.

Our current priorities for MCNP are quality, value, and new features.

“Quality” encompasses not only rigorous quality control but also demonstrations of the code’s validity and accuracy by comparison with experiment. “Value” encompasses good documentation, timely and controlled releases of code versions, and sufficient standardization and portability that the code can be run on the computers favored by MCNP users. “New features” are the essence of code development but cannot supersede quality and value in importance.

The Future

The future of the Monte Carlo method is secure. As computers become cheaper and faster, it will more and more become the method of choice for solving a wide range of problems. And as the applications of radiation increase, Monte Carlo radiation-transport codes will become more and more widely used. The future of MCNP at the Laboratory is also bright, despite reductions in nuclear-weapons research. In fact, the code is likely to become a more important tool, as physical testing of nuclear weapons is phased out and simulations come to constitute a larger fraction of weapons research. And every enhancement in MCNP benefits not only the Department of Energy’s defense programs but also its programs in many other areas: nuclear-criticality safety, environmental restoration, nuclear-waste management, prevention of nuclear proliferation, accelerator production of tritium, accelerator transmutation of nuclear waste, space nuclear power, nuclear safeguards, fusion and fission energy, arms control, intelligence, and on and on. Furthermore, MCNP is one of the Laboratory’s most promising candidates for technology transfer to industry. Many companies have already expressed interest in Co-

operative Research and Development Agreements on utilizing or jointly developing MCNP for specific applications. Those collaborations and contacts can provide a wide range of industries with a marvelous introduction to the Laboratory. ■

Acknowledgements

Sabrina illustrations provided by Kenneth Van Riper. MCNP is the product of many people over many years, and the following list of present and past contributors to its development is bound to be incomplete. Unless otherwise noted, all are or were affiliated with Los Alamos National Laboratory. Present principal developers: Thomas E. Booth, Judith F. Briesmeister, R. Arthur Forster, John S. Hendricks, H. Grady Hughes, Gregg W. McKinney, Richard E. Prael. Recent significant developers: Thomas N. K. Godfrey, Leland L. Carter (Westinghouse Hanford), David G. Collins, Guy P. Estes, Henry Lichtenstein, Robert C. Little, Shane P. Pederson, Roger R. Roberts, Robert G. Schrandt, Robert E. Seamon, Edward C. Snow, Patrick D. Soran, William M. Taylor, William L. Thompson, Kenneth Van Riper, Laurie S. Waters, James T. West III (IBM), William T. Urban. Graduate students and other contributors: Ronald C. Brockhoff, Robert S. Brown, David A. Cardon, Peter P. Cebull, David E. Hollowell, J. Steven Hansen, Glen T. Nakafuji, Scott P. Palmtag, Everett L. Redmond II, James E. Sisolak, Jennifer L. Uhle, Todd J. Urbatsch, Jason W. VanDenburg, John C. Wagner, Daniel J. Whalen. Distinguished “alumni”: Edmond D. Cashwell, C. J. Everett, Enrico Fermi, Nicholas C. Metropolis, John von Neumann, Robert Richtmyer, Stanislaw Ulam.

Further Reading

- N. Metropolis and S. Ulam. 1949. The Monte Carlo method. *Journal of the American Statistical Association* 44: 335–341.
- W. W. Wood. 1986. Early history of computer simulations in statistical mechanics. In *Molecular-Dynamics Simulation of Statistical-Mechanical Systems. Proceedings of the International School of Physics Enrico Fermi, course 97*, edited by G. Ciccotti and W. G. Hoover. North-Holland Physics Publishing.
- Roger Eckhardt. 1987. Stan Ulam, John von Neumann, and the Monte Carlo method. *Los Alamos Science* 15: 131–137. Los Alamos National Laboratory.

N. Metropolis. 1987. The beginning of the Monte Carlo method. *Los Alamos Science* 15: 125–130. Los Alamos National Laboratory.

L. L. Carter and E. D. Cashwell. 1975. *Particle-Transport Simulation with the Monte Carlo Method*. U.S. Energy Research and Development Administration. Available as TID-26607 from National Technical Information Service, Springfield, Virginia 22161.

K. Binder, editor. 1984. *Applications of the Monte Carlo Method in Statistical Physics*. Topics in Current Physics, volume 36. Springer-Verlag.

R. A. Forster, R. C. Little, J. F. Briesmeister, and J. S. Hendricks. 1990. MCNP capabilities for nuclear well logging calculations. *IEEE Transactions on Nuclear Science* 37: 1378–1385.

Ivan Lux and Laszlo Koblinger. 1991. *Monte Carlo Transport Methods: Neutron and Photon Calculations*. CRC Press.

J. F. Briesmeister, editor. 1993. MCNP: A general Monte Carlo N-particle transport code, version 4A. Los Alamos National Laboratory report LA-12625.



John S. Hendricks received his B.S. and M.S. in nuclear engineering from the University of California, Los Angeles, in 1972 and his Ph.D. in nuclear engineering from the Massachusetts Institute of Technology in 1975. He joined the Laboratory in 1975 as a staff member in the Radiation Transport Group and has remained with that group ever since, except for a two-year stint on the staff of the Laboratory’s director as the Congressional liaison. He is currently team leader for Monte Carlo in the Radiation Transport Group.



State-of-the-Art Parallel Computing

molecular dynamics on the connection machine

Peter S. Lomdahl and David M. Beazley

This "snapshot" from a molecular dynamics simulation shows a thin plate undergoing fracture. The plate contains 38 million particles. This state-of-the-art simulation was performed using a scalable algorithm, called SPaSM, specifically designed to run on the massively parallel CM-5 supercomputer at Los Alamos National Laboratory. The velocities of the particles are indicated by color—red and yellow particles have higher velocities than blue and gray particles.

The availability of massively parallel computers presents computational scientists with an exciting challenge. The architects of these machines claim that they are scalable—that a machine containing 100 processors arranged in a parallel architecture can be made 10 times more powerful by employing the same basic design but using 1000 processors instead of 100. Such massively parallel computers are available, and one of the largest—a CM-5 Connection Machine containing 1024 processors—is here at Los Alamos National Laboratory. The challenge is to realize the promise of those machines—to demonstrate that large scientific problems can be computed with the advertised speed and cost-effectiveness.

The challenge is significant because the architecture of parallel computers differs dramatically from that of conventional vector supercomputers, which have been the standard tool for scientific computing for the last two decades.

The massively parallel architecture requires an entirely new approach to programming. A computation must somehow be divided equally among the hundreds of individual processors and communication between processors must be rapid and kept to a minimum. The incentive to work out these issues was heightened by Gordon Bell, an independent consultant to the computer industry and well-known skeptic of massively parallel machines. Six years ago he initiated a formal competition for achievements in large-scale scientific computation on vector and parallel machines. The annual prizes are awarded through and recognized by the Computer Society of the Institute of Electrical and Electronic Engineers (IEEE). The work reported here was awarded one of the 1993 IEEE Gordon Bell prizes. We were able to demonstrate very high performance as measured by speed: Our simulation, called SPaSM (scalable parallel short-range molecular dynamics), ran on the massively parallel CM-5 at a rate of 50 gigaflops (50 billion floating-point operations per second). That rate is nearly 40% of the theoretical maximum; typical programs achieve only about 20% of the theoretical maximum. Also, the algorithm we developed has the property of scalability. When the number of processors used was doubled, the time required to run the simulation was cut in half.

Our achievement was the adaptation of the molecular-dynamics method to the architecture of parallel machines. Molecular dynamics (MD) is a first-principles approach to the study of materials. MD starts with the fundamental building blocks—the individual atoms—that make up a material and follows the motions of the atoms as they interact with each other through interatomic forces. Even the smallest piece of material that is useful for ex-

perimental measurement contains over ten billion atoms. Therefore, the larger the number of atoms included in the calculation, the more interesting the results for materials science. Scientists would very much like to perform simulations that predict the motions of billions of atoms during a dynamical process such as machining or fracture and compare the predictions with experiment. Unfortunately, the realization of that prospect is not possible today. However, if the machines can be scaled up in both memory capacity and speed, as the computer manufacturers have promised, then simulations can be scaled up proportionally so that billion-atom simulations may become a reality. Furthermore, as we show here, the current generation of massively parallel machines has made it possible to simulate the motions of tens of millions of atoms. Such simulations are large enough to perform meaningful studies of the bulk properties of matter and to improve the approximate models of those properties used in standard continuum descriptions of macroscopic systems. Molecular-dynamics simulations on massively parallel computers thus represent a new opportunity to study the dynamical properties of materials from first principles.

Massively parallel machines have only recently become popular, so the vendors have not yet developed the software—the compilers—needed to optimize a given program for the particular architecture of their machines. Consequently, to develop a scalable parallel algorithm for molecular dynamics that would yield high performance, we needed to understand the inner workings of the CM-5 and incorporate that knowledge into the details of our program. Much of our work focused on implementing a particular technique for interprocessor communication known as “message passing.” In this

approach programmers use explicit instructions to initiate communication among the processors of a parallel system, and the processors accomplish the communication by sending messages to each other.

Message-passing programming is advantageous not only in achieving high performance, but also in increasing “portability,” because it is easily adapted for use on a number of massively parallel machines. We were able to run our message-passing algorithm on another parallel computer, the Cray T3D, with minimal modification.

Following brief descriptions of MD and the architecture and properties of the CM-5, we present the method by which we have mapped a molecular-dynamics simulation to the architecture of the CM-5. Our example illustrates message passing and other techniques that are widely applicable to effective programming of massively parallel computers.

Molecular Dynamics

Molecular dynamics is a computational method used to track the positions and velocities of individual particles (atoms or groups of atoms) in a material as the particles interact with each other and respond to external influences. The motion of each particle is calculated by Newton’s equation of motion, $\text{force} = \text{mass} \times \text{acceleration}$, where the force on a given particle depends on the interactions of the particle with other particles. The mutual interaction of many particles simultaneously is called a “many-body problem” and has no analytical, or exact, solution because the force on a particle is always changing in a complex way as each particle in the system moves under the influence of the others.

The MD algorithm provides an ap-

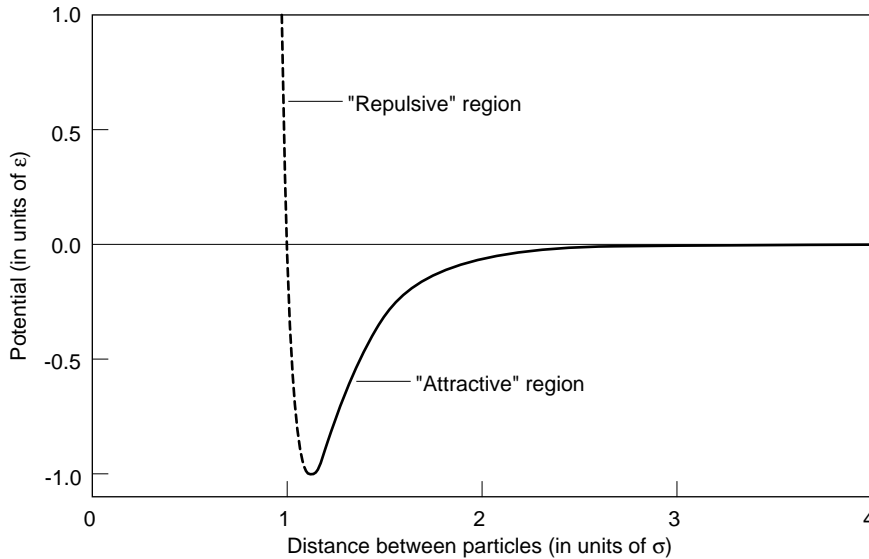


Figure 1. The Lennard-Jones Interaction Potential

The Lennard-Jones interaction potential V between two particles is plotted as a function of the distance r between the particles. The potential is in units of ϵ , and the distance between the particles is in units of σ . The parameter σ is equal to the value of r at which the potential function crosses zero. The potential has a positive slope at longer distances, corresponding to an attractive force, and a steeply negative slope at short distances, corresponding to a strong repulsive force. Separating the two regions is the local minimum of the potential—the deepest point in the potential “well”—which has a depth equal to ϵ and is located at $r = 2^{1/6}\sigma$. At the potential minimum the force between the particles is zero. Therefore, in the absence of effects from any other particles, the separation between two particles will tend to oscillate around $r = 2^{1/6}\sigma$.

proximate solution to the many-body problem by treating time as a succession of discrete timesteps and treating the force on (and therefore the acceleration of) each particle as a constant for the duration of a single timestep. At each timestep the position and velocity of each particle is updated on basis of the constant acceleration it experiences during that timestep. For example, at time t , at the beginning of a timestep of length Δt , particle i has position \mathbf{r}_i and velocity \mathbf{v}_i . The force on particle i from the other particles is calculated; the force determines the acceleration \mathbf{a}_i of particle i for the duration of this timestep. From elementary physics, the position \mathbf{r}_i' and velocity \mathbf{v}_i' at the later time $t + \Delta t$ are given by:

$$\mathbf{r}_i' = \mathbf{r}_i + \mathbf{v}_i\Delta t + \frac{1}{2}\mathbf{a}_i(\Delta t)^2$$

and

$$\mathbf{v}_i' = \mathbf{v}_i + \mathbf{a}_i\Delta t.$$

Sometimes, these equations are replaced by more sophisticated approximations for the new positions and velocities; the approximations involve interpolations from positions and velocities at time t and the earlier $t - \Delta t$. Such schemes conserve energy and momentum more effectively than the simple update of position and velocity given above.

Whatever the scheme, the equations yield a new position and velocity for each particle at the new time $t + \Delta t$.

The procedure is repeated for each succeeding timestep, and the result is a series of “snapshots” of the positions and velocities of the particles at times $t=0$, Δt , $2\Delta t$, ..., $n\Delta t$.

In MD simulations particles are usually treated as point particles (that is, they have no extent), and the force between any two particles is usually approximated as the gradient of a potential function V that depends only on the distance r between the two particles. The force on particle i from particle j can be written

$$F_{ij} = -\frac{dV}{dr}\bigg|_{r_{ij}}, \text{ where } r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|,$$

and that force is directed along the line connecting the two particles. If the force is negative, particles i and j attract one another; if the force is positive, the particles repel each other. The net force exerted on particle i by all other particles j is approximated by summing the forces calculated from all pair-wise interactions between particle i and each particle j .

The standard pair-wise potential used in MD simulations is the Lennard-Jones potential, which is known empirically to provide a reasonably good “fit” to experimental data for atom-atom interactions in many solids and liquids. As shown in Figure 1, this potential has both an attractive part and a repulsive part:

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right],$$

where r is the distance between the two particles and σ and ϵ are adjustable parameters. When the distance between two particles is at the minimum of the potential function, the force between the two particles is zero and the parti-

cles are in their equilibrium positions. In the absence of other particles nearby, the separation of two particles will tend to oscillate around this distance. Physically, the average nearest-neighbor distance of atoms in many solids and fluids corresponds very closely to the position of the minimum in the atom-atom potential.

For our molecular-dynamics simulations, we simplify the force calculation greatly by using a truncated form of the Lennard-Jones potential. That is, we “cut off” the long, flat tail of the potential and assume that the force between particles is exactly zero for particle separations greater than some chosen r_{\max} , the interaction cut-off distance. In other words, when calculating the forces for particle i , we need consider only those particles j that are within a distance r_{\max} of i . The time required to calculate the pair interactions with all the surrounding particles is reduced considerably since a given particle “feels” forces from perhaps only a hundred neighbors instead of the millions or billions of particles included in the simulation. The truncated potential provides an excellent approximation for solid materials because in solids the effects of distant atoms are “screened” by nearer atoms. The fact that we can use a short-range potential is the most significant difference between our problem and that described in “A Fast Tree Code for Many-Body Problems,” in which the force is long-range and therefore each particle always interacts with all other particles in the system.

Mapping Molecular Dynamics onto the Architecture of the CM-5

The molecular-dynamics method has been implemented on electronic computers since 1957. The first such calcu-

lations were performed on the UNIVAC, the first commercially available computer. Initially only 32 particles were used, and only about 300 interactions per hour could be calculated. The rapid advance of computers has made it possible for us—using the massively parallel CM-5—to include tens of millions of atoms in our simulation of materials behavior. Since materials contain large numbers of repeating units (units that can be treated in parallel) calculations of materials behavior are ideally suited for implementation on massively parallel architectures.

Figure 2 illustrates the architecture of the CM-5, which was built by Thinking Machines Corporation. The computer was constructed from large numbers of relatively inexpensive, high-volume microprocessor and memory components. Such a scheme allows the hardware designers to build a high-performance supercomputer without a major effort in the development of new microprocessor and memory circuits. The Laboratory’s CM-5 has 1024 processing nodes but partitions as small as 32 nodes may be used.

The processing nodes in the CM-5 are connected by two types of high-speed communication networks: a control network and a data network. The control network performs synchronization functions, broadcasts, and reductions. The synchronization functions are used to coordinate the work of the individual processors. Broadcasts are messages sent to all processors simultaneously, such as a set of instructions to be executed by each processor. Reductions are operations in which data are collected from all processors for the calculation of some overall value such as the total energy of a system.

The data network carries simultaneous point-to-point communication between multiple processing nodes at a rate of over 5 megabytes per second per

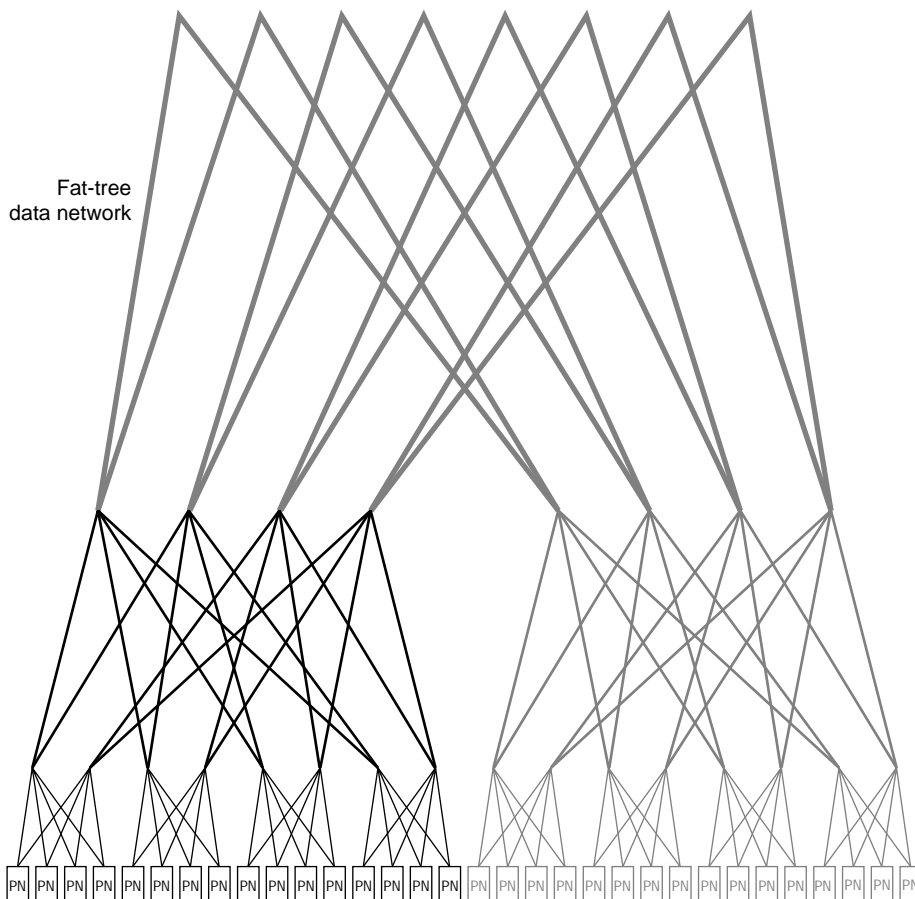
node. Both networks have a “fat-tree” topology, a scalable structure in which the total bandwidth (capacity for data transmission) of the network increases in proportion to the number of processors. When new processing nodes are added, the networks are expanded so that the effective bandwidth between any two processors does not decrease. Maintaining a high bandwidth is critical to the scalability of the CM-5. A machine having more processing nodes will have a greater total amount of network “traffic.” Therefore the larger computer must also have a greater network capacity in order to realize the expected gain in performance.

Our MD algorithm uses the data network extensively—large amounts of particle data are transmitted between processors. The control network is also very important to our algorithm. A single program is broadcast over the control network to the processors, each of which executes that program for the appropriate subset of the particles in the simulation. The processing nodes are allowed to operate independently during most of the calculations. However, there are steps in our algorithm that require synchronization of the processors. For example, the calculation of new particle positions and velocities cannot proceed until the total force acting on each particle has been determined. A special synchronization function ensures that the latter calculation is complete before the former is initiated.

Access to the communication networks is provided by a message-passing library supplied by Thinking Machines. The library allows two types of communication—synchronous and asynchronous. We use both communication styles in our MD algorithm.

All programs on the CM-5 use some form of message passing between processing nodes. Data-parallel languages such as C* and FORTRAN-90 perform

(b) 32-node CM-5



(a) 16-node CM-5

(c) Processing node (PN)

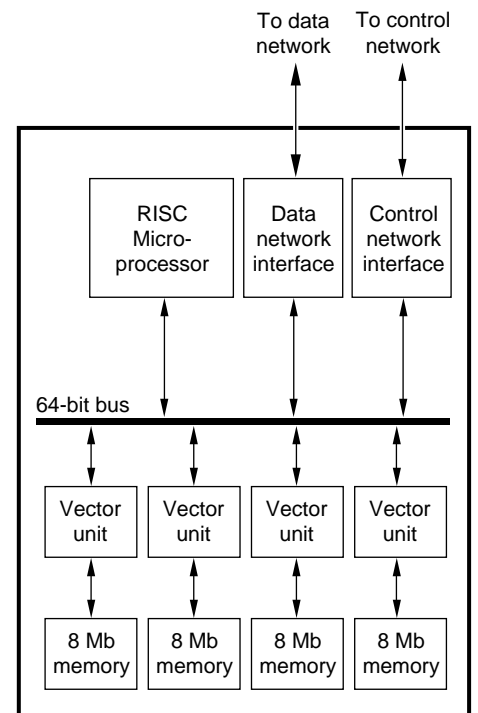


Figure 2. The Architecture of the CM-5 Connection Machine

The diagram shows the general layout of the CM-5, one of the largest massively parallel computers. Also shown in the diagram is the manner in which this parallel architecture can be scaled up in size. (a) Sixteen processing nodes are connected in parallel by a network that carries data between the processors. A control network (not shown), similar in structure to the data network, carries instructions to the nodes. The networks have a "fat tree" topology—the upper branches (which connect a large number of nodes and therefore must carry more data) have a higher bandwidth (capacity for data transmission) so that the overall point-to-point bandwidth is at least 5 megabytes per second for any pair of nodes. (b) The addition of more processing nodes to the system (shown in gray) requires that the networks be expanded to a "higher" level in order to maintain the point-to-point bandwidth. (c) Each processing node consists of a 33-megahertz SPARC RISC (Reduced Instruction Set Computer) microprocessor, 32 megabytes of memory, and four vector-processing units that perform 64-bit floating-point and integer arithmetic at a maximum combined rate of 128 million floating-point operations per second. The 1024-node CM-5 at the Laboratory also has 400 gigabytes (400 billion bytes) of disk space distributed over a number of disk drives (not shown), which are connected in parallel to the networks by input/output processors. The parallel arrangement of the disk drives allows data to be transferred rapidly to disk; however, the data of a single write operation will be spread over all disks. For more details of computer elements and architectures see "How Computers Work: An Introduction to Serial, Vector, and Parallel Computers."

message passing transparently; that is, the underlying communication is hidden from the user. However, to achieve higher performance we wrote explicit message-passing instructions into our program. It was designed to be executed on all of the processing nodes simultaneously. Once the program is broadcast to the processing nodes, each node works independently on a small part of the problem and manages its own data and interprocessor communications.

Data structures of the MD algorithm. The particles in a molecular-dynamics simulation occupy a simulated region of space called a “computational box,” shown in Figure 3a (for simplicity, the algorithm is illustrated in two dimensions). The box is divided into domains of equal size, one for each processor (Figure 3b). The assignment of particles to processors is made according to the domain in which each particle is located. The data associated with each particle include its velocity and its coordinates within the computational box.

Because any simulation is finite in size we often need a method to effectively eliminate the boundaries of the simulation. Therefore, we apply periodic boundary conditions to the computational box. That is, the entire computational box with all of its particles is treated as just one unit of an infinite lattice of identical units. Thus a particle near a boundary of the computational box will be treated as if it is interacting with particles in an adjacent box containing an identical number of particles with identical positions and velocities. One result of such boundary conditions is that when a particle moves out through one side of the computational box, an identical particle with the same velocity enters through the boundary at the opposite side of the box. Therefore, both linear momentum and

the number of particles in the computational box are conserved.

Typically the volume of the computational box is large enough that the processor domains have dimensions significantly larger than the interaction cut-off distance, r_{\max} . In such cases each processor will have been assigned a significant number of particles that do not interact with each other. Computing the forces between all pairs of particles within each domain is unnecessary because the result for each pair of particles separated by more than r_{\max} would be zero. Therefore, the domain of each processor is subdivided into an identical number of cells, each having dimensions slightly larger than r_{\max} (see Figure 3c). The number of such cells depends only on the size of the domain and r_{\max} , not on the number of available processors. The domains may be subdivided into thousands of cells for simulations with large spatial dimensions. In the example in Figure 3, there are 4 processor domains and each domain has 16 cells. The cell structure forms the foundation of our algorithm.

Each particle is represented in the computer by a C-programming-language data structure consisting of the particle parameters, including position, velocity, force, and type. Associated with each cell is a small block of memory containing a sequential list of the particle data for that cell. Storing particle data in this manner facilitates the communication steps of the force calculation. The entire contents of a cell can be communicated to other processors simply by sending a small block of memory.

MD force calculation. The calculation of forces acting on the particles is the most time-consuming part of the MD method. To optimize the calculation, particles that are neighbors physically should also be near one another in

the memory of the computer. However, such an arrangement is difficult to maintain because each particle is free to move anywhere in space—particles that are initially neighbors may separate during the course of an MD simulation, and particles that are initially far apart may become neighbors. The cell structure just described was designed to keep the particles organized so that the force calculation proceeds efficiently.

The force on a given particle includes contributions from all the other particles that are closer than r_{\max} . Because the cell size has been chosen to be slightly larger than r_{\max} , all the particles that must be considered are located within the cell containing the given particle or within adjacent cells.

Each processor follows an identical, predetermined sequence to calculate the forces on the particles within its assigned domain. For each cell in its domain, the processor computes forces exerted on particles in the cell by other particles in the same cell as well as by particles located in adjacent cells. The forces due to particles in adjacent cells are calculated in the order given by the interaction path shown in Figure 4a. Use of an identical path for each cell ensures that all contributions to the total force on every particle are computed without performing redundant calculations.

Typical calculations have hundreds or thousands of cells per processor domain, so for most of the cells of a given domain all neighboring cells are also assigned to the same domain. Therefore, the interaction calculation can be performed for most cells without any communication between processors. However, for cells along the edge of each processor domain, some of the adjacent cells are assigned to other processors (as in Figure 4b), and the message-passing features of the CM-5 must be utilized so that particle data for

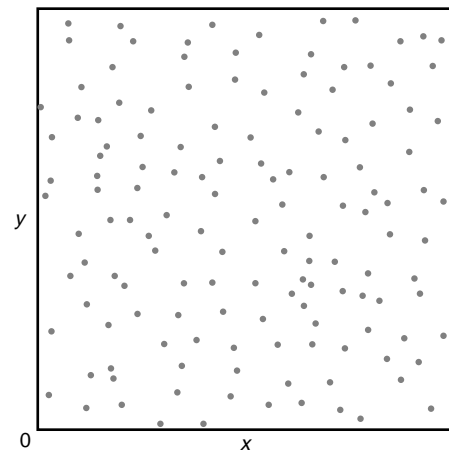
those adjacent cells can be sent to and received from the neighboring processors. Because each processor has been assigned an identical cell structure and follows the same sequence of operations through that structure, all processors will be required to transmit particle data for the same internal cell at approximately the same time. At such times the processors synchronize and participate in send-and-receive communication. Processors assigned to domains with fewer particles will proceed through the interaction calculations faster. Therefore, when working on cells at the edges of the domain, those processors may be required to wait for others before they can synchronize. The details of synchronized message passing are shown in Figure 4c.

At every message-passing step, each processor sends the particle data for an entire cell to the appropriate processors and receives corresponding information from other processors. The force calculation then proceeds, now that each processor has available in its local memory the particle data needed to implement the next step along the interaction path. Note that synchronization occurs only during message passing. At all other times, the processors are running asynchronously.

Redistribution of particles. When the force calculation is complete the processors compute the new positions and velocities of all particles in the system. Since our algorithm is based on the cell structure of the processor domains, the possibility of changes in particle positions requires that the computer data structures for each cell are updated regularly. A special redistribution function checks the coordinates of each particle after every timestep. If the new coordinates of a particle indicate that it has moved to a new cell, the particle parameters must be transferred

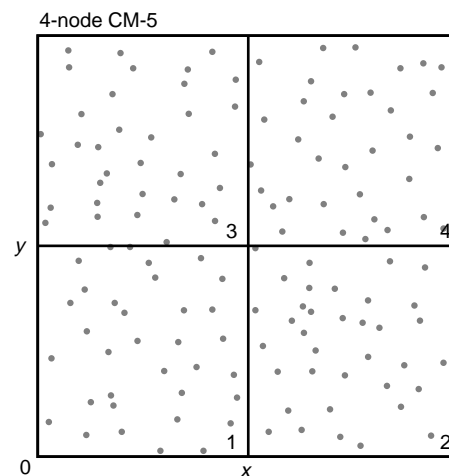
Figure 3. The Data Structure of the MD Algorithm

This figure illustrates the mapping of a molecular-dynamics problem onto a parallel processor with 4 nodes. For simplicity, the algorithm is illustrated in two dimensions.



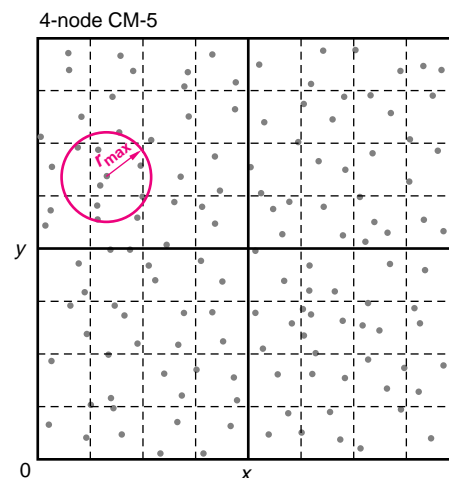
(a) The computational box

A “computational box” contains all the particles in the simulation. A single coordinate system is used for the entire computational box so that each particle has a unique set of coordinates.



(b) The processor domains

The computational box is divided into 4 equal domains, each of which is assigned to a separate processor as denoted by the numbers 1, 2, 3, and 4.

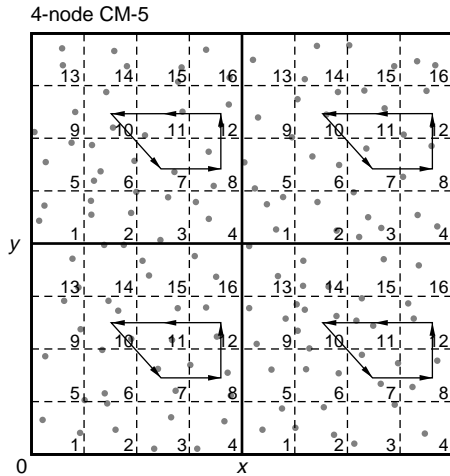


(c) The cell structures of each processor domain

The domain assigned to each processor is subdivided into cells having dimensions just larger than r_{\max} , the interaction cut-off distance. In this example the domain of each processor has been divided into 16 cells. The circle of radius r_{\max} is centered on a particle and indicates the area containing all the other particles that are close enough to affect the motion of the center particle. Note that all the particles contained in this circle are located either within the same cell as the center particle or within adjacent cells. Associated with each cell is a small block of memory containing a sequential list of the particle data for that cell.

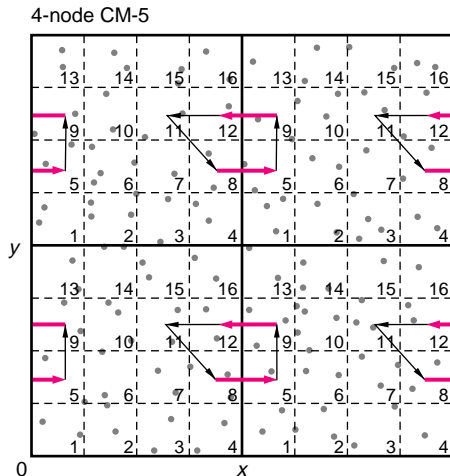
Figure 4. MD Force Calculation

This figure shows the steps that are performed for each cell in order to calculate and sum up the forces acting on all particles. Each processor works on its cells in the order 1 through 16 and follows an identical interaction pathway for each cell.



(a) Interaction path for cell 7

The processors are performing the interaction calculations for cell 7, having already done so for cells 1 through 6. First, all forces between particles within cell 7 are computed and the results are sent to an accumulator that adds all the pair-wise contributions for each particle. Then the block of memory containing the list of particle data for cell 8 is called up and used to calculate the forces exerted on particles in cell 7 by particles in cell 8. The results are sent to the accumulator for cell 7. By Newton's third law, the forces exerted on particles in cell 8 by particles in cell 7 are equal and opposite to those exerted on particles in cell 7 by particles in cell 8. Consequently, the results are also sent to the accumulator for cell 8. The process is repeated three times more for cell 7, calculating forces involving particles in cell 7 with those in cells 12, 11, and 10. The forces exerted on particles in cell 7 by particles in cells 2, 3, 4, and 6 were sent to the accumulator for cell 7 when the interaction pathways were followed for cells 2, 3, 4, and 6. Because all the particles that must be considered for cell 7 are within the domain of a single processor, no message passing is required.



(b) Interaction path for cell 8 (synchronous message passing steps are red)

The processors are calculating forces for cell 8. Because cell 8 is located at the edge of a processor domain, some steps of the interaction path (shown by red arrows) cross the boundary between processor domains. To carry out the entire force calculation for cell 8 (and for all other cells along the edge of processor domains) the data for the particles in cell 8 must be made available to neighboring processors. Synchronous message passing is used to transfer the particle data for cell 8 to and from neighboring processors. Note that because the computational box has periodic boundary conditions, interaction paths leave the box at one edge while equivalent interaction paths enter the box at the opposite edge. Because each processor has the same cell structure and proceeds through the cells in the same order, all the processors will need to send and/or receive particle data for a given edge cell at approximately the same time. Those processors that are assigned to neighboring domains in the computational box will exchange data with one another during the force calculation. Therefore, synchronization of message passing between the appropriate processors is straightforward.

(c) Synchronous message passing between processing nodes (PN)

One processor is sending data to another via synchronous message passing. The source processor stops computing, issues a "Ready to send" signal, and waits for a "Ready to receive" signal from the destination processor. After the "Ready to receive" signal is issued, the data is transferred, and then both processors resume calculations.

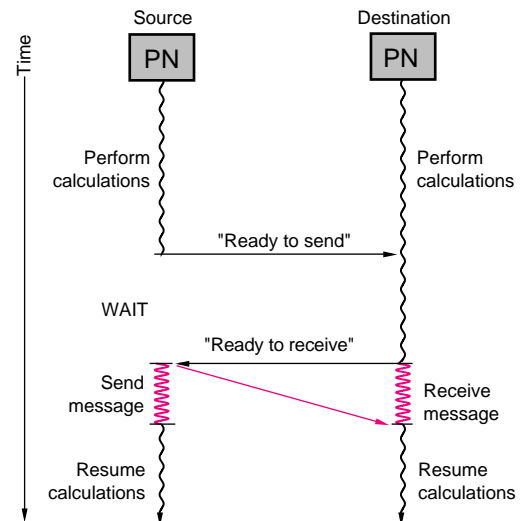
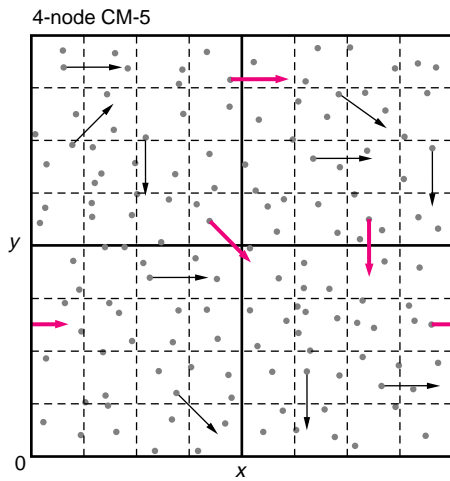


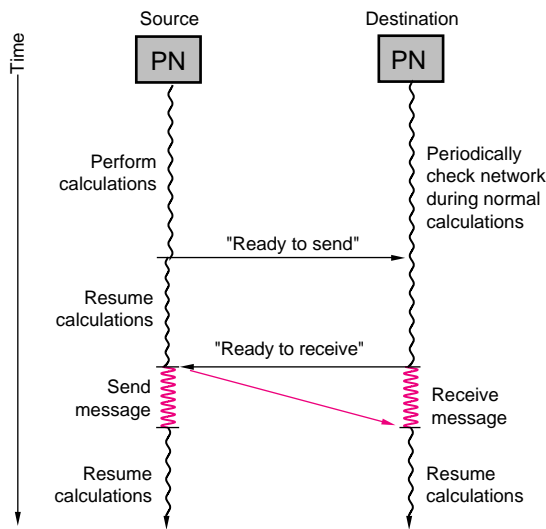
Figure 5. Redistribution of Particles

After the force calculation is completed for a given timestep, new positions and velocities are calculated for all the particles.



(a) Moving particles to proper cells (asynchronous message passing steps are red)

The new coordinates of each particle are checked by a special redistribution function. For each case in which the new coordinates of a particle correspond to a location in a different cell, the redistribution function also transfers the data for that particle to the sequential list of particle data for the new cell. Most of the transfers are between cells within the same processor domain. Those transfers that cross a boundary between processor domains are executed by means of message passing between processing nodes and are indicated by red arrows in the figure.



(b) Asynchronous message passing between processing nodes (PN)

Since there is no way to know beforehand which processors will be involved in the transfer of particles, the transfer is accomplished by an asynchronous mode of message passing. During asynchronous message passing, the source processor issues a "Ready to send" signal and immediately resumes calculation (contrast with synchronous message passing in Figure 4c). All processors periodically check the network and retrieve any waiting messages. The only significant interruption to the calculations of either node is the time used for the transfer of the message.

Particles	Processors					
	32	64	128	256	512	1024
1,024,000	8.90	4.51	2.32	1.26	0.72	0.44
2,048,000	—	8.96	4.44	2.46	1.36	0.74
4,096,000	—	—	8.79	4.81	2.67	1.36
8,192,000	—	—	16.83	8.81	4.80	2.47
16,384,000	—	—	—	16.95	8.74	4.49
32,768,000	—	—	—	—	16.90	8.54
65,536,000	—	—	—	—	—	16.55
131,072,000	—	—	—	—	—	34.26

Table 1. Update Times per Timestep in Scaling Simulations
 Shown in the table are the timing results for a set of test simulations ($r_{\max} = 2.5\sigma$). Each entry in the table is the CPU time (in seconds) required to compute a single timestep for the given combination of numbers of particles and processors.

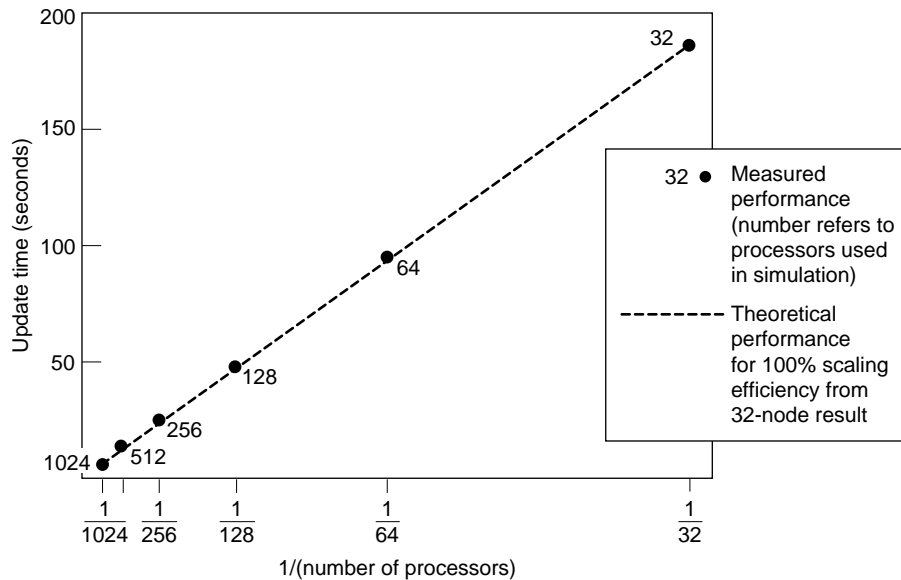


Figure 6. The Scalability of the MD Algorithm
 The figure is a plot of the CPU time required to simulate a single timestep versus the reciprocal of the number of processors used for the calculation. The dots represent the results obtained from a 4-million-particle simulation ($r_{\max} = 5\sigma$) in which particles rearranged from a simple cubic lattice to a face-centered cubic lattice. The line represents perfect scaling from the 32-node data point, that is, an update time that is inversely proportional to the number of processing nodes used in the simulation. The results of this series of test calculations indicate that the performance of our algorithm, at least for this type of simulation, scales very well with the number of available processors.

to the particle list of the new cell (see Figure 5a).

There are two types of particle transfers to consider. One involves the simple transfer of a particle from one cell to another within a single processor domain. In this case, deleting particle data from one cell and adding it to a new cell requires only copying of memory contents within a single processing node. The second type of transfer requires moving particle data to a cell list in another domain, so the particle data is deleted from the original cell list and sent to the new processor by message passing. The redistribution process cannot use synchronized message passing since it is not known how many particles will leave each processor, how many will be received by each processor, and from what processors particles will arrive. Therefore all processors are put into an asynchronous message-passing mode. Asynchronous message passing occurs in the background while the particle coordinates are checked (see Figure 5b). The messages are addressed so that they go to the appropriate processor, and each node periodically checks the network and receives any messages that are waiting. Messages may be sent or received at any time, and, in general, all processors operate independently. The redistribution of particle data is complete when all particle coordinates have been checked and all messages have been retrieved from the data network.

Our experience indicates that the redistribution of particles is efficient and accounts for a small portion of the overall iteration time. Generally, the number of particles changing cells after any given timestep is small compared with the total number of particles in the system. In addition, since most of the particles that change cells simply move to another cell on the same processor, the inter-processor communication re-

quired for the redistribution procedure is minimal.

The architecture and capabilities of the CM-5 are well suited for molecular-dynamics applications. The high-speed communication networks allow us to use message passing to efficiently transfer data among the processors. An equivalent domain was assigned to each processor in order to divide the workload among the processors, and the cell structure and interaction path ensure that the interaction calculations are completed in a minimum amount of time.

The Scalability of our MD Algorithm

In ideal circumstances, the speed of a well-designed parallel algorithm will scale linearly with the number of processors used in the parallel computer. To test the scalability of our algorithm we repeatedly performed a test simulation using various numbers of processors and various numbers of particles ranging from 1 million to 131 million. The starting condition for the tests consisted of identical particles arranged in a simple cubic lattice. The simple cubic lattice—known to be unstable for an interaction potential that depends only on r and to undergo a phase change in which the particles rearrange to a face-centered cubic configuration—was chosen to guarantee movement of particles between domains during the calculation.

Table 1 shows the results of the scaling tests in which r_{\max} was set equal to 2.5σ . Each doubling of the number of processors used for a given number of particles approximately halved the update times (the CPU time required to compute all particle motions during a single timestep). For a 4-million-particle simulation in which r_{\max}

was equal to 5σ , the increase in speed in going from 32 processors to 1024 processors was more than a factor of 30, corresponding to 95% parallel efficiency. The latter scaling results are shown graphically in Figure 6, where the update times for the 4-million-particle simulation are plotted as a function of the reciprocal of the number of processors used. The scalability of our MD algorithm was also demonstrated by comparing the update times for varying numbers of particles (see Table 1). For a given number of processors, the update times increased linearly with the number of particles in the simulation.

Applications

As a demonstration of the practicality of our algorithm, we performed a simulation of a 1-million-particle projectile colliding with a 10-million-particle plate. Figure 7 is a series of images taken from the simulation; each image is a “snapshot” showing the positions and velocities of the particles at the end of a timestep. The velocities are indicated by color—red and yellow particles are moving faster than blue and gray particles. The top panel in Figure 7 shows the system at timestep 1000, just after the projectile has made contact with the plate. Note that the particles in the lower third of the projectile have higher kinetic energies (higher velocities) than those in the remainder of the projectile. Some of the particles in the plate also have increased kinetic energies. At timestep 2000 (middle panel of Figure 7) the projectile has nearly penetrated the plate and part of the projectile has begun to break up. A shock wave has propagated to the edges of the plate. By timestep 2900 (bottom panel of Figure 7) part of the projectile has been absorbed into the plate, the remainder of the projectile has disinte-

grated into free particles, and the shape of the plate has become significantly distorted.

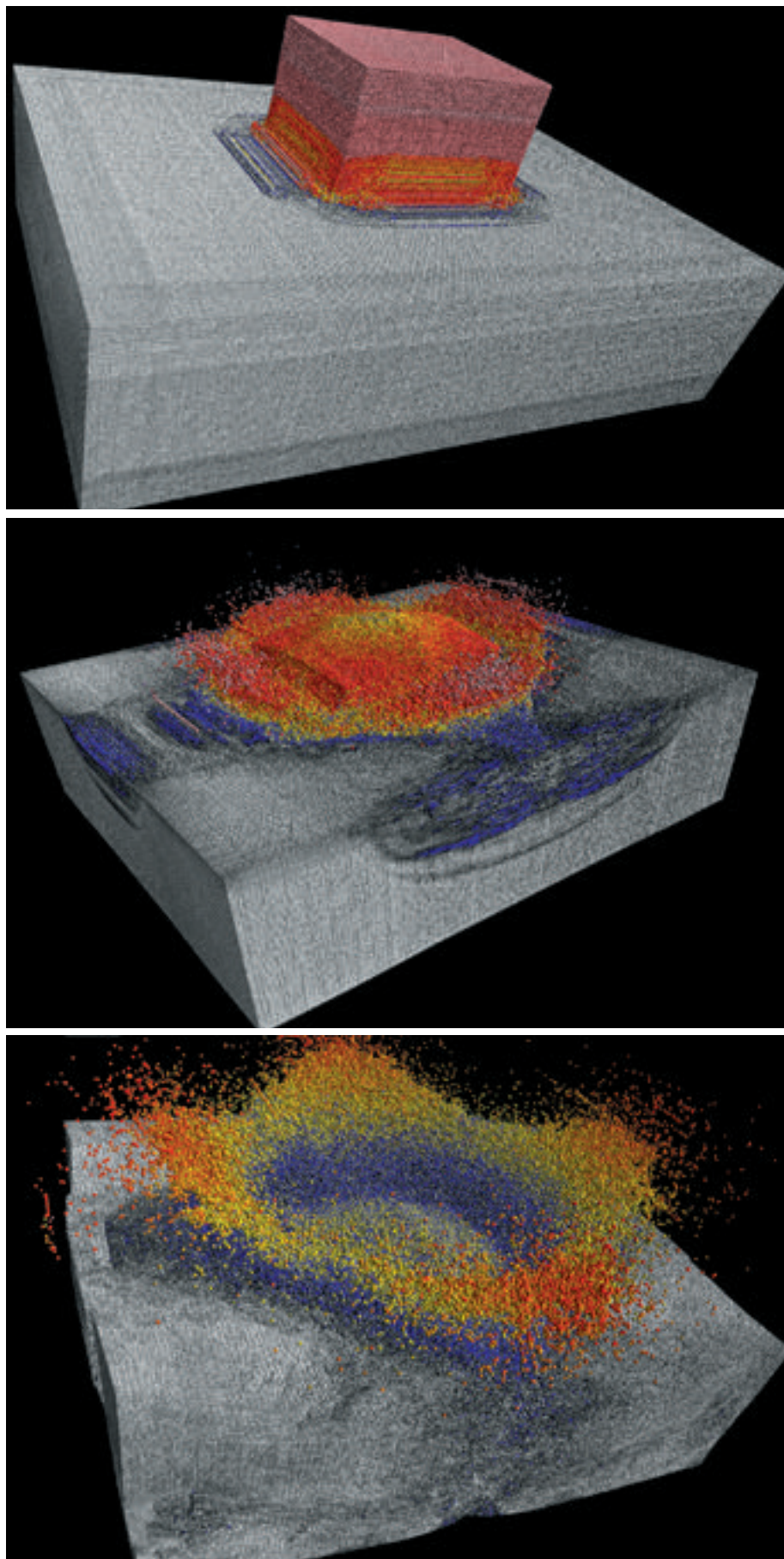
The impact simulation shows the inherently unstructured nature of MD simulations. Tracking the motions of individual particles often requires significant amounts of communication and data management. We have been able to achieve high performance by carefully mapping the physics of the problem to the architecture of the CM-5. The fast iteration times we have attained allow the calculation of larger MD simulations than have previously been possible. For example, the illustration at the beginning of this article was taken from one of the largest MD simulations performed to date—our 38-million-particle simulation of a plate undergoing fracture. Because our algorithm is scalable, the size of the physical system that can be modeled is expected to increase proportionally as each new generation of massively parallel machines is developed.

We are now working to incorporate new types of interaction potentials into our MD algorithm. The particular atomic interactions will be described by materials-specific interaction potentials like the “embedded-atom method” potential. Such potential functions are created by adding a “many-body” term to the pair-interaction potential presented in this paper. The many-body term varies according to the local density surrounding a given atom and thereby results in a more physically realistic interaction potential.

We are using our MD algorithm to study the physics of fracture. We are particularly interested in understanding the brittle and ductile behavior of materials. In ductile fracture the propagation of cracks is accompanied by the formation of a significant number of dislocations (discontinuities in the crystalline structure of a material) at the

Figure 7. An 11-Million-Particle Impact Simulation

The three images show the progress of an MD simulation in which a 1-million-particle projectile impacts a 10-million-particle plate. The particles of both the projectile and the plate were initially at rest in equilibrium positions in face-centered cubic lattices. The interactions between particles were approximated by using a Lennard-Jones potential. The colors in the figure represent the velocities of the particles; red and yellow particles have higher velocities than blue and gray particles. The simulation of 2900 timesteps required 30 hours of CPU time on a 512-processor CM-5. The top, middle, and bottom panels correspond to timesteps 1000, 2000, and 2900, respectively.



crack tip. These dislocations are responsible for permanent deformations in the material. When the fracture is brittle, the crack propagates with little or no permanent deformation. Our goal is to understand and ultimately control parameters that influence these phenomena because such knowledge and ability will allow us to design materials having specific responses and behavior. Large-scale molecular dynamics is an ideal tool for the investigation of fracture phenomena because MD allows us to perform simulations of near-macroscopic samples using very few simplifications or approximations. ■

Acknowledgements

It is a pleasure to acknowledge the contributions of Niels Grønbech-Jensen, Pablo Tamayo, and Brad L. Holian to parts of the work described here. We also acknowledge generous support from the staff of the Advanced Computing Laboratory, who provided both machine access and assistance. In particular, David O. Rich helped with the use of the CM-5, and Michael F. Krogh provided invaluable help with visualization.

Further Reading

M. P. Allen and D. J. Tildesley. 1987. *Computer Simulations of Liquids*. Clarendon Press.

D. M. Beazley and P. S. Lomdahl. 1994. Message-passing multi-cell molecular dynamics on the Connection Machine 5. *Parallel Computing* 20: 173–195.

D. M. Beazley, P. S. Lomdahl, P. Tamayo, and N. Grønbech-Jensen. 1994. A high performance communications and memory caching scheme for molecular dynamics on the CM-5. In *Proceedings of the Eighth International Parallel Processing Symposium*. IEEE Computer Society.

R. C. Giles and P. Tamayo. 1992. A parallel scalable approach to short-range molecular dy-



David M. Beazley (left) is a graduate student in the Center for Nonlinear Studies and the Condensed Matter and Statistical Physics Group. His research interests include massively parallel supercomputing, high-performance computer architecture, and scientific computing. Beazley began working at the Laboratory while an undergraduate in 1990. He received a B.A. in mathematics from Fort Lewis College in 1991, and is currently working on a Ph.D. in computational science at the University of Utah.


Peter S. Lomdahl is a staff member in the Condensed Matter and Statistical Physics Group in the Theoretical Division, where he has worked on computational condensed-matter and materials-science research since 1985. From 1982 to 1985 he was a postdoctoral fellow with the Center for Nonlinear Studies. Lomdahl received his M.S. in electrical engineering and his Ph.D. in mathematical physics from the Technical University of Denmark in 1979 and 1982. His research interests include parallel computing and nonlinear phenomena in condensed-matter physics and materials science.

namics on the CM-5. In *Proceedings of Scalable High Performance Computing Conference 1992*. IEEE Computer Society.

P. S. Lomdahl, P. Tamayo, N. Grønbech-Jensen, and D. M. Beazley. 1993. 50 GFlops molecular dynamics on the connection machine. In *Proceedings of Supercomputing 1993*. IEEE Computer Society.

S. Plimpton. 1993. Fast parallel algorithms for short-range molecular dynamics. Sandia National Laboratory Report SAND91-1144, UC-705.

P. Tamayo, J. P. Mesirov, and B. M. Boghosian. 1991. Parallel approaches to short range molecular dynamics simulations. In *Proceedings of Supercomputing 1991*. IEEE Computer Society.



Cold dark matter, illustrated here in blue and black, is shown forming halos around galaxies and connecting the giant collection of galaxies known as the Great Wall. The picture is an artist's conception, as cold dark matter has never been observed. Its existence is called for by one theory of how gravity caused the growth of large-scale structures such as those pictured here. The latest simulation "experiments" on massively parallel computers have sufficient speed and accuracy to test the cold-dark-matter scenario.

EXPERIMENTAL COSMOLOGY

*and the puzzle of large-scale
structures* Wojciech H. Zurek and Michael S. Warren

A critical limitation in astrophysics is the impossibility of testing theories with controllable and repeatable experiments. Cosmologists face the worst version of this problem. Not only are the data restricted to observations of uncontrolled events, but also the experiment was performed only once—there is only one universe! Moreover, only a fraction of the universe that is in principle observable (given the finite speed of light and the finite age of the universe) is accessible in practice to observation. Cosmology has suffered from having too little hard data and too much freedom to build theoretical models, so that even the most basic questions—about the size of the universe, the age of the universe, the kinds of matter in the universe—are still only partially answered.

This situation is changing through advances in observation and computation. Instruments such as the Hubble Space Telescope and the Cosmic Background Explorer satellite are providing new data, some relating to the very early history of the universe, that help constrain cosmological theories. Furthermore, computer simulations carried out on the most advanced massively parallel machines now contain enough physics and are sufficiently accurate to predict detailed consequences of the many proposed models. Here we present high-resolution computer simulations that address one of the outstanding puzzles in modern cosmology: How did the observed distribution of galaxies—the so-called large-scale structure of the universe—arise?

Cosmologists once thought that the matter distribution on large scales is fairly uniform, that galaxies like ours are sprinkled evenly throughout the cosmos. Over the last few decades, however, observations out to distances of billions of light-years* have revealed a “froth-like” structure to the universe. Large numbers of galaxies are grouped into clusters, and those clusters appear to be interconnected by thin sheets and filaments of galaxies surrounding large low-density cells, or “voids,” approximately a hundredth of the radius of the universe in size. Figure 1 illustrates some of these large-scale structures, from individual galaxies with bright components on the order of 10 kiloparsecs in radius (a parsec is about 3.3 light-years), to clusters of galaxies on scales of 1 to 10 megaparsecs, and finally to sheets or filaments of galaxy

clusters that surround large voids on scales of 30 to 50 megaparsecs. By comparison, the current size of the observable universe is over a hundred times larger, roughly 10,000 megaparsecs across.

In confronting the data on large-scale structure, the task of a cosmologist is somewhat similar to the task of a prosecutor trying to prove that the accused (a cosmological model) is indeed guilty of the crime (the creation of the large-scale structure), while his observational colleagues are trying either to support the case by supplying the observational evidence (which is nearly always circumstantial) or to side with the defense and provide the alibi. The crime has, of course, happened only once, and over the years many suspects have been rounded up. It is conceivable, though, that the guilty party has not yet come under suspicion. However, many possible models have already been exonerated.

Consider the idea that nearly all the matter in the universe is contained in visible, luminous stars and that gravity is the force primarily responsible for the clumping of matter into structures of various sizes. After all, that idea does indeed describe our solar system. But on galactic and larger scales, that natural model has been “found innocent.” As we shall see below, it cannot account for the internal dynamics of spiral galaxies. Instead those galaxies must contain about ten times more mass than has so far been observed to explain the rapid motion of the stars inside them.

More important for our discussion, it is difficult to explain how gravitational forces alone produced large-scale structure unless we assume the presence of still greater amounts of invisible mass—what is usually referred to as dark matter. It seems that there is neither enough mass in the luminous matter

nor enough time since the beginning of the universe for primordial fluctuations in the density of that matter to have grown under the influence of gravity into the structures now observed. The size of the primordial fluctuations, as inferred from inhomogeneities in the cosmic background radiation, is just not large enough. Thus, observations on galactic and larger scales present clear evidence that the universe contains much more mass than “meets the eye.”

What is this dark matter that dominates the mass content of the universe? Astrophysicists have responded to this puzzle with a variety of ideas. Some postulate that the dark matter is more or less ordinary (that is, made of the same stuff as our sun and the planets) but, for some reason, simply invisible. Calculations of the primordial synthesis of the light elements, however, place an upper limit on the amount of ordinary matter that might be present, a limit that is probably too low for ordinary matter alone to explain large-scale structure. Other astrophysicists suggest that neutrinos with a small, but non-negligible, mass are the main ingredient of dark matter. The latter postulate leads to the so-called hot-dark-matter (HDM) models: Massive but not-too-heavy neutrinos remain relativistic or “hot” (they move with velocities comparable to the velocity of light), until fairly late in the history of the universe. Their movements wipe out density perturbations on galactic scales and thereby determine a certain predictable course for the development of structure, which turns out to be too slow to match observations.

Perhaps the best-defined model, and the one we have tested through computer simulation, is the so-called cold-dark-matter (CDM) model. The main ingredient of dark matter in the CDM model is not specified beyond the requirements that it be very weakly interacting with ordinary matter and suffi-

*These distances and all the other absolute distances in this article are uncertain by approximately a factor of 2 for various reasons, primarily the uncertainty in the Hubble parameter. Relative distances can however be expressed in a manner independent of those uncertainties by using the redshift, which will be introduced in the following section.

(a) Harvard-Smithsonian survey of spatial distribution of galaxies.

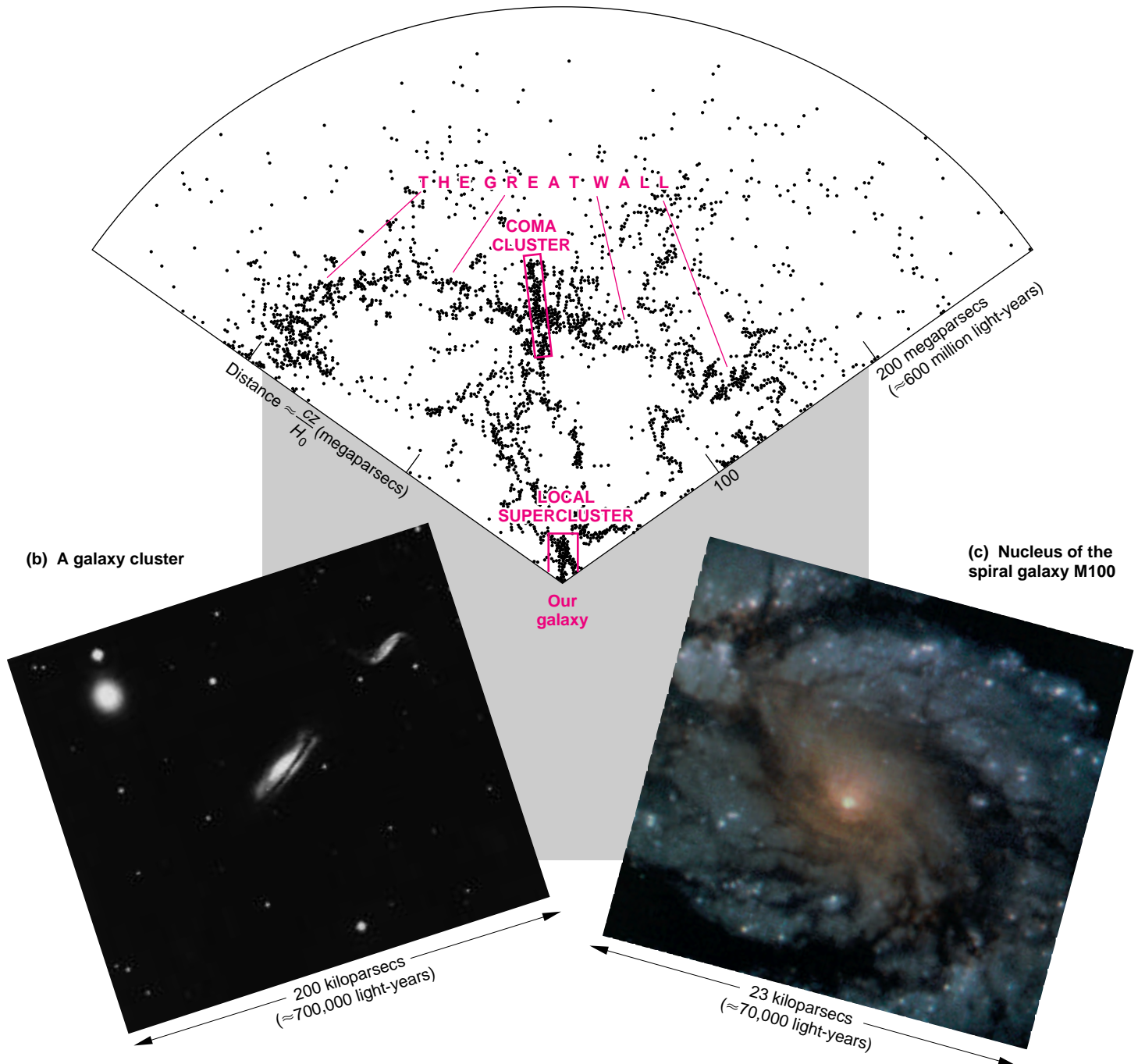


Figure 1. Inhomogeneity of the Universe

The distribution of matter in the universe is marked by clumps and voids on all scales. (a) The spatial distribution of galaxies according to a survey by the Harvard-Smithsonian Center for Astrophysics. Each point represents a galaxy. The distance from the vertex (our galaxy) to the outer edge of the region surveyed is roughly 300 megaparsecs. The distances plotted are inferred from an observed measure of distance called redshift, z , through the use of Hubble's Law—distance = v/H_0 , where $v \approx cz$ for objects with small redshifts—as will be discussed later in the main text. Here c is the velocity of light and H_0 is Hubble's constant. The elongated clump of galaxies in the center of the figure includes the rich cluster in the constellation Coma Berenices. It is part of the "Great Wall," the large sheet of galaxies running across the image at intermediate distances, which includes more than half the galaxies in the figure. The Great Wall and the voids beside it are among the largest known structures in the universe. (Figure adapted by permission of Margaret J. Geller, Harvard-Smithsonian Center for Astrophysics.) (b) A galaxy cluster—a structure of intermediate size—in the constellation Leo. (Photograph courtesy of Palomar/California Institute of Technology.) (c) The nucleus of a single spiral galaxy, catalogued as M100, with a radius on the order of 10 kiloparsecs. Galaxies are the smallest objects we consider in our study of large-scale structure. (Photograph courtesy of NASA/Space Telescope Science Institute.)

ciently massive to become “cold,” or nonrelativistic (moving at velocities small compared with the velocity of light), very early in the history of the universe, before the universe was about a thousand years old. The model assumes, as an initial condition, a simple, scale-independent spectrum of primordial density fluctuations in the cold dark matter that are postulated to grow under the influence of gravity to form the large-scale structures that we now see.

The only important free parameter in the CDM model is the size of the assumed primordial density fluctuations. Information about those fluctuations is imprinted in the cosmic background radiation, the oldest detectable remnant of the early universe. That sea of microwave photons, which fills all space, is thought to have remained essentially unscattered since the time when radiation decoupled from matter and the universe became transparent to radiation. It would therefore have retained inhomogeneities present at the time of decoupling, when, according to standard Big Bang Cosmology, the universe was about 10,000 years old.

The size of those inhomogeneities was recently determined by data from the Cosmic Background Explorer satellite. Known as COBE (pronounced to rhyme with Toby) the satellite is dedicated to measuring various properties of the cosmic background radiation. The 1992 COBE data show variations in the temperature of the cosmic background radiation of about one part in 10^5 on distance scales of around 1 billion light years (hundreds of megaparsecs), depending on the spatial direction of the measurements. The small temperature fluctuations are a direct measure of the amplitudes of the fluctuations in the matter density present when the cosmic background originated. Thus the size of the temperature differences measured

by COBE fixes the size of the primordial density fluctuations in the CDM model.

Our contribution has been to incorporate the new COBE data into the initial conditions of the CDM model and then determine, through high-resolution, state-of-the-art simulations, the CDM predictions for the growth of structure through time. Our simulations keep track of the long-range gravitational forces among 17 million point masses with a precision sufficient to resolve density contrasts of six orders of magnitude on scales that differ by as much as four orders of magnitude. Our program has undergone continual development since it received the 1992 Gordon Bell Prize for practical parallel-processing research. It is described in “A Fast Tree Code for Many-Body Problems,” immediately following this article.

The high resolution that we have achieved has allowed much more detailed comparison with observations than was previously possible. Our results discussed on page 78 suggest that the CDM model, recently abandoned by many, should still be taken seriously.

Before presenting those comparisons, we will explain the initial conditions of the simulations and the basic assumptions of the CDM model through a review of standard Big Bang cosmology. The CDM model and all other recent models of structure formation are defined within that basic framework, a framework that has been strengthened by all recent observations.

The Expansion of the Universe

Big Bang cosmology grew out of the most far-reaching and well-accepted cosmological observation: The universe is expanding. In other words, to any observer in the universe, distant objects such as galaxies appear to be re-

ceding from the observer at velocities v proportional to l , the distance from the object to the observer. This relation, called Hubble’s law, was discovered by Edwin Hubble in 1929. It is written

$$v = H_0 l.$$

The Hubble “constant,” H_0 , is shorthand for $H(t_0)$, which means the value of the Hubble parameter $H(t)$ measured now, at $t = t_0$. Although Hubble’s law was deduced from observations carried out from our vantage point, the planet Earth, it is generalized to the cosmos by assuming that at any given time the universe looks qualitatively the same to all observers regardless of their locations and the directions in which they look. This assumption, called the Cosmological Principle, is fundamental to the study of cosmology.

Figure 2 shows a traditional way of visualizing the expansion of the universe. The three-dimensional universe is replaced by an analogous two-dimensional surface, the surface of a sphere. As the sphere expands, any observer confined to the surface sees other points receding with speeds proportional to their distances from the observer, as observed by Hubble.

Hubble discovered this proportionality, and thus the expansion of the universe, by measuring both the apparent velocities v of nearby galaxies and their distances l . Measuring the distances to galaxies is difficult, involving a ladder of distance measurements from nearby stars (tens of parsecs away) to more distant stars in our own galaxy (up to 10 kiloparsecs away) to stars in nearby galaxies (1 to 10 megaparsecs away) and on to more distant galaxies. The process remains problematic, so that even today uncertainties in the distance measurements yield estimates of H_0 ranging from 50 to 100 kilometers per second per megaparsec. A convenient

way to represent this uncertainty is to define a parameter h such that $H_0 = h \cdot 100 \text{ (km/s)/Mpc}$, where $1/2 \leq h \leq 1$ (Mpc is the abbreviation for a megaparsec). To ensure that our calculations do not assume a universe too young to be consistent with other predictions of astrophysics, we have taken the value of h to be $1/2$, or the value of H_0 to be 50 (km/s)/Mpc , throughout this article and in our simulations.*

In contrast to the distance, the apparent velocity of a receding galaxy is reliably and easily determined by measuring the Doppler shift—in this case the redshift—of the light that it emits. More specifically, one identifies the absorption and emission lines in the spectrum of the galaxy and measures the shift of those features toward longer wavelengths, in the direction from blue toward red. (Here again is an application of the Cosmological Principle: One is assuming that the matter (atoms) and the wavelengths of the light it emits are the same throughout the universe.) The size of the redshift z is related to the apparent recession velocity v through the Doppler-shift formula:

$$1+z \equiv \frac{\lambda_{\text{observed}}}{\lambda_{\text{emitted}}} = \left(\frac{1+v/c}{1-v/c} \right)^{1/2},$$

in which λ_{emitted} is the wavelength emitted by the galaxy, $\lambda_{\text{observed}}$ is the wavelength of the observed signal, and c is the speed of light. For galaxies with small redshifts, or with small recession velocities $v \ll c$, the redshift $z \approx v/c$, or the recession velocity is given by

$$v \approx zc.$$

Thus the velocities of nearby galaxies

*The new corrective optics recently installed in the Hubble Space Telescope should help put the controversy about the value of H_0 to rest by skipping several of the rungs in the ladder of distance measurements.

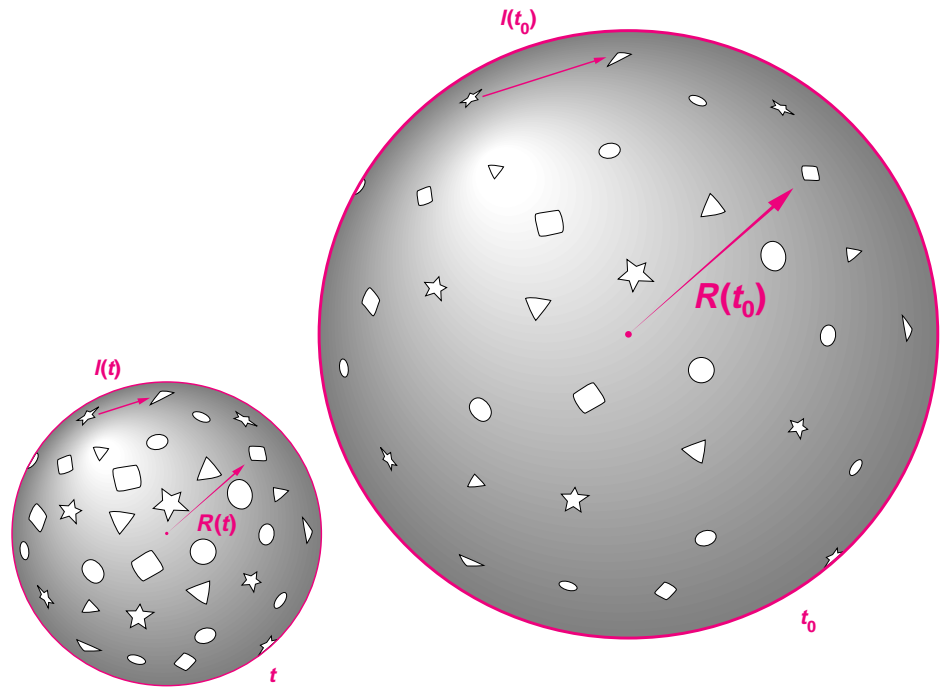


Figure 2. The Expansion of the Universe

The Hubble flow and the expansion of the universe are depicted here in a spherical fragment of the real universe. The size of the sphere, $R(t)$, changes with time, as do the distances $l(t)$ between the objects (such as galaxies) inside. If all of the objects started expanding from the same point at different velocities v , then their present locations will depend on how fast they are moving. This dependence yields a proportionality relation, $l = v/H$, which is the content of Hubble's law. The velocity of a galaxy can usually be inferred from the redshift, z , of the light emitted by it, and when $z \ll 1$ the velocity is given by $v = zc$. Because the light travels at a finite velocity, the redshift is also a measure of the "look-back time," which in turn is related to the relative size of the universe at the time when the photons were emitted: Expansion "stretches" the wavelengths of the photons and the size of the universe at the same rate, so that the photons emitted at the instant t and detected now, at t_0 , will be redshifted—the photon wavelengths will be longer by the redshift factor $1 + z = R(t_0)/R(t)$. It is convenient to define a universal scale factor $a(t)$ —the factor by which all of the distances in the real universe need to be rescaled to account for the effect of the universal expansion. Thus $R(t_0)/R(t) = l(t_0)/l(t) = a(t_0)/a(t)$. By definition $a(t_0) = 1$. Thus $a(t) = 1/[1 + z(t)]$.

are directly proportional to their observed redshifts. The formula applies even to the most distant galaxies shown in Figure 1. They have redshifts of 0.05 and by Hubble's law are at distances of $l \approx zc/H_0$, or 300 megaparsecs for $h = 1/2$.

In contrast, the most distant object observed to date is a quasar with a redshift of nearly 5. The full Doppler-shift formula implies that that object is rushing away from us at the enormous velocity of $0.95c$ or about 285,000 kilometers per second, and, by Hubble's law, is at a distance of approximately 5700 megaparsecs.

In general, the higher the redshift of an object, the greater is its distance from us, the faster is its recession velocity, and the longer its signal took to reach us. The observed redshift of an object thus indicates the "look-back time," a fact that is used again and again to interpret cosmological observations and to piece together the history of the universe.

Cosmologists typically relate the redshift not to the look-back time but rather to the relative size of the universe, because the latter relationship is simpler. The redshift of a light signal (more exactly, $1 + z$) is inversely pro-

portional to the relative size, or scale, of the universe at the time that signal originated. Here the size of the universe is measured by the distances between nearby objects, $l(t)$, or by the radius, $R(t)$, of a spherical fragment of the universe. As depicted in Figure 2, both quantities increase with the universal expansion. The relative size—the universal scale factor—is then defined as the ratio

$$a(t) \equiv l(t)/l(t_0) \equiv R(t)/R(t_0).$$

In other words, the universal scale factor is defined to be one at present, $a(t_0) = 1$, and its value decreases to zero as we go back in time.

The inverse proportionality between the redshift and the universal scale factor can be derived by reinterpreting the origin of the redshift as due to the overall expansion of the universe. As the universe expands, the distances between objects co-moving with the universal expansion increase proportionally to $R(t)$. In addition, the wavelengths of photons traversing the universe at velocity c are stretched by the expansion in proportion to $R(t)$. The ratio of wavelengths in the redshift formula can therefore be written as

$$\frac{\lambda_{\text{observed}}}{\lambda_{\text{emitted}}} = \frac{R(t_0)}{R(t)},$$

where t_0 is now, the time at which the signal is observed, and t is the time at which the signal originated. Rewriting this equation in terms of the universal scale factor shows that the redshift increases as the scale factor decreases:

$$1 + z(t) = \frac{R(t_0)}{R(t)} = \frac{1}{a(t)}.$$

This equation means, for example,

that when we observe a quasar with a redshift of 5 (or $1 + z = 6$), we are looking back in time to when distances between astronomical objects were approximately $1/(1 + z) = 1/6$ of their present values. According to standard Big Bang cosmology, the cosmic background photons, which contain the oldest imprint of large-scale structure, have been redshifted by a factor of about 1000 ($z \approx 1000$), which means that those photons decoupled from matter at the time when the scale factor was only one-thousandth of its present size. Our simulations of the clumping of matter into large-scale structures begin later, at $z \approx 100$, or when the scale factor was approximately one-hundredth of its present size.

The Big Bang and the Definition of Ω

Extrapolating the expansion of the universe all the way back in time suggests that initially any point was arbitrarily close to any other point, or $R(t=0) \approx 0$ and $a(t=0) \approx 0$. Such a picture is consistent with solutions to the equations that arise when general relativity is applied to the entire cosmos; it is also consistent with observations. Therefore cosmologists now generally believe that the universe expanded and cooled from an initial state of extremely high temperature and density. Cosmologists can trace the history back no farther than the time when the density of the universe was the so-called Planck density (roughly 5×10^{33} grams/centimeter³). At that density quantum-gravity effects begin to dominate. The initial explosive expansion is called the Big Bang.

How much time has elapsed since the Big Bang? We need to estimate the age of the universe to determine the time available for the development of

large-scale structure. The simplest estimate is made by assuming that the Hubble parameter has remained constant at its present value of H_0 and thus the speed of expansion has also remained constant since the Big Bang. Then t_0 , the time since the Big Bang, is approximated by

$$t_0 \approx \frac{\text{distance expanded}}{\text{speed of expansion}}$$

and, by Hubble's law, that time is

$$t_0 \approx \frac{1}{H_0} \approx 20 \text{ billion years}$$

where we have taken the value of the Hubble constant to be 50 (km/s)/Mpc.

The true age of the universe must be somewhat less than 20 billion years because the initial expansion at $t = 0$ must have slowed down over time due to the mutual gravitational attraction of the total energy content, or equivalent mass content, in the universe. The theory of relativity relates all forms of energy E , even pure radiation, to an equivalent mass m through the formula $E = mc^2$. Thus both matter and radiation are sources of gravitation and contributed to the slowing down of the initial expansion. As the universe has expanded, the Hubble parameter $H(t)$ has been decreasing continuously to its present value H_0 .*

The time history of the expansion is typically described in terms of the Hubble parameter where

*For $H_0 = 50$ (km/s)/Mpc and $\Omega = 1$ (see below), the age of the universe is 14 billion years, approximately the age of the oldest globular clusters inferred from stellar evolution and primordial abundances of the light elements. A larger value of H_0 , or equivalently h , would imply a younger universe and could thus lead to a contradiction, or at the very least, indicate a low value for Ω .

Figure 3. Evolution of the Universe

Each curve shows a possible history of the size of the universe as a function of time, depending on whether the average equivalent mass density $\bar{\rho}$ of the universe is greater than, equal to, or less than the critical value, ρ_{critical} , or equivalently on whether Ω is greater than, equal to, or less than 1. The curves can be obtained by using the Newtonian approximation to predict the motion of a spherical fragment of the universe of radius $R(t)$ and containing mass M equal to $4\pi R^3 \bar{\rho}/3$. The Newtonian approximation can be derived from Einstein's equations when $GM/Rc^2 \ll 1$, that is, when $R(t)$ is large compared to the Schwarzschild radius of the mass M . Then the radius of the sphere satisfies the familiar equation

$$\frac{d^2 R}{dt^2} = -\frac{GM}{R^2}, \quad (1)$$

which can be used in turn to derive the equally familiar statement of conservation of energy in a gravitating system:

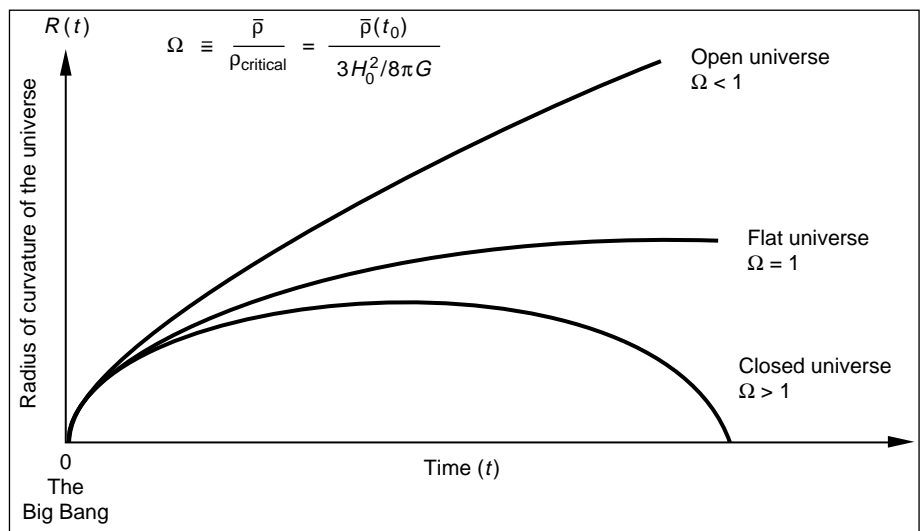
$$\left(\frac{dR}{dt}\right)^2 = \frac{2GM}{R} + C. \quad (2)$$

The constant C on the right-hand side appears in the course of the integration involved in the transition from the first equation to the second. Its sign determines the balance between the kinetic energy of expansion (proportional to the left-hand side of this equation) and the potential energy (on the right-hand side). By dividing both sides of equation 2 by R^2 (to express the right-hand side in terms of the density $\bar{\rho}(t)$ rather than mass M) and re-expressing $R(t)$ as $R_0 a(t)$, one eventually arrives at the equation

$$H(t)^2 \equiv \frac{1}{a^2} \left(\frac{da}{dt}\right)^2 = \frac{8\pi G \bar{\rho}(t)}{3} - \frac{kc^2}{R^2}. \quad (3)$$

Here k is the constant C from equation 1, rescaled so that it can assume only the values ± 1 and 0. It is known as the curvature constant. It should be emphasized that the above derivation is perfectly correct, and not merely a Newtonian analogue: The condition $R \gg GM/c^2$ can always be satisfied by adopting a sufficiently small radius.

Each curve is a solution to the equation obtained by using one of the different values of k (and by assuming that matter, not radiation, makes the dominant contribution to the energy density). Regardless of the value of k and the composition of the universe, solutions to the equation have $R = 0$ at $t = 0$; that is, they entail a Big Bang at the beginning of the universe. Also, the equation implies that $k = +1$ if $\Omega \equiv \bar{\rho}(t)/\rho_{\text{critical}}(t) > 1$, where $\rho_{\text{critical}}(t) \equiv 3(H(t))^2/8\pi G$. Likewise $k = 0$ if $\Omega = 1$, and $k = -1$ if $\Omega < 1$. The value of k , or equivalently the value of Ω , determines whether the universe is open, flat, or closed, as explained in the main text.



$$H(t) \equiv \frac{1}{a(t)} \frac{da}{dt} = \frac{1}{R(t)} \frac{dR}{dt},$$

and $a(t)$ is the universal scale factor introduced above. The relevant differential equation from general relativity is presented in the caption for Figure 3. This equation yields three different types of universe, depending on the average density of the universe, or more exactly, on the value of Ω , the dimensionless density parameter. This parameter is defined as the ratio of the average equivalent mass density in the universe, $\bar{\rho}(t)$, to a critical density:

$$\Omega \equiv \frac{\bar{\rho}(t)}{\rho_{\text{critical}}(t)}$$

where

$$\rho_{\text{critical}}(t) \equiv \frac{3(H(t))^2}{8\pi G},$$

and G is Newton's gravitational constant. Figure 3 shows the three possible histories of the expansion of the universe depending on whether the value of Ω is greater than, equal to, or less than one.

In Newtonian cosmology, which applies when most energy is in the form of matter, Ω has a simple interpretation. It is just the ratio of the magnitude of the gravitational potential (or binding) energy of matter to the kinetic energy of the universal expansion:

$$\Omega = \frac{|E_{\text{potential}}|}{E_{\text{kinetic}}}.$$

Thus, if the potential energy is less than the kinetic energy ($\Omega < 1$), galaxies have enough energy to escape the pull of gravity and will travel to infinity. Alternately, if the potential energy is greater than the kinetic energy ($\Omega >$

1), galaxies are gravitationally bound, the Hubble expansion will eventually stop, and the universe will contract back on itself.

The same criterion defines the dividing line between two entirely different types of universes allowed by general relativity. As shown in Figure 3, if $\Omega < 1$, the universe is said to be open; it is spatially infinite and will continue expanding forever. If $\Omega > 1$, the universe is said to be closed; it is spatially finite (being curved like the sphere in Figure 2) and will eventually stop expanding and begin to contract [$H(t)$ will eventually become negative]. A third solution also exists. Namely, if Ω is exactly equal to 1, the universe is said to be flat; it is infinite in space and time, but differs from an open universe in that the recession speeds of galaxies eventually approach zero rather than a positive constant.*

Observational evidence suggests that Ω should lie within the generous bounds $0.2 < \Omega < 2.0$. This range is obtained from observations that usually allow one to estimate Ω more directly than by comparing the present average density of the universe, $\bar{\rho}(t_0)$, with the critical density:

$$\rho_{\text{critical}}(t_0) \equiv \frac{3H_0^2}{8\pi G} \\ \sim 10^{-29} h^2 \text{ g/cm}^3.$$

This number, uncertain by a factor of 4 because of the uncertainty in the value of the Hubble constant, corresponds to the mass density of a few hydrogen atoms per cubic meter.

Determinations of Ω tend to yield larger values when they employ observations on larger scales. Thus Ω

*Note that if Ω is equal to 1 at any time, it remains constant at that value. Otherwise its value changes with time, but because the total energy of the universe is conserved, Ω cannot change from being less than 1 to being greater than 1 or vice-versa.

inferred from galactic scales tends to be on the order of 0.1 or less, while clusters of galaxies (scales of a few megaparsecs) indicate an Ω of 0.2 or more. On still larger scales of 30 to 50 megaparsecs, Ω as large as 1 is necessary to explain coherent flows of galaxies such as the "Great Attractor."

Cosmologists must assume a value for Ω to investigate the development of structure in the expanding universe. Although the present density of luminous matter, $\bar{\rho}_{\text{luminous}}(t_0)$, appears to be less than one percent of the present critical density, that is $\Omega_{\text{luminous}} < 0.01$, most theorists—for both esthetic and theoretical reasons—take the case of the flat universe ($\Omega = 1$) very seriously. In that case, most of the mass in the universe must be in the form of dark matter.

Formation of Structure in a Cold-Dark-Matter Universe

A major event in the history since the Big Bang was the shift from the radiation-dominated era, when the universe was so hot that most of the energy was in the form of radiation rather than matter, to the matter-dominated era, when the universe had cooled down enough that most of the energy was in the form of matter. The main constituents of matter then were the two most stable baryons, namely protons and neutrons, enough electrons to balance the charge on the protons, and perhaps, as proposed by the CDM model, an exotic brand of noninteracting, cold, dark matter. [Note that during the first few minutes after the Big Bang, nuclear synthesis reactions had caused most neutrons to combine with protons (hydrogen nuclei) to form helium nuclei.] The transition to a matter-dominated universe (described in "Big Bang Cosmology and the Microwave

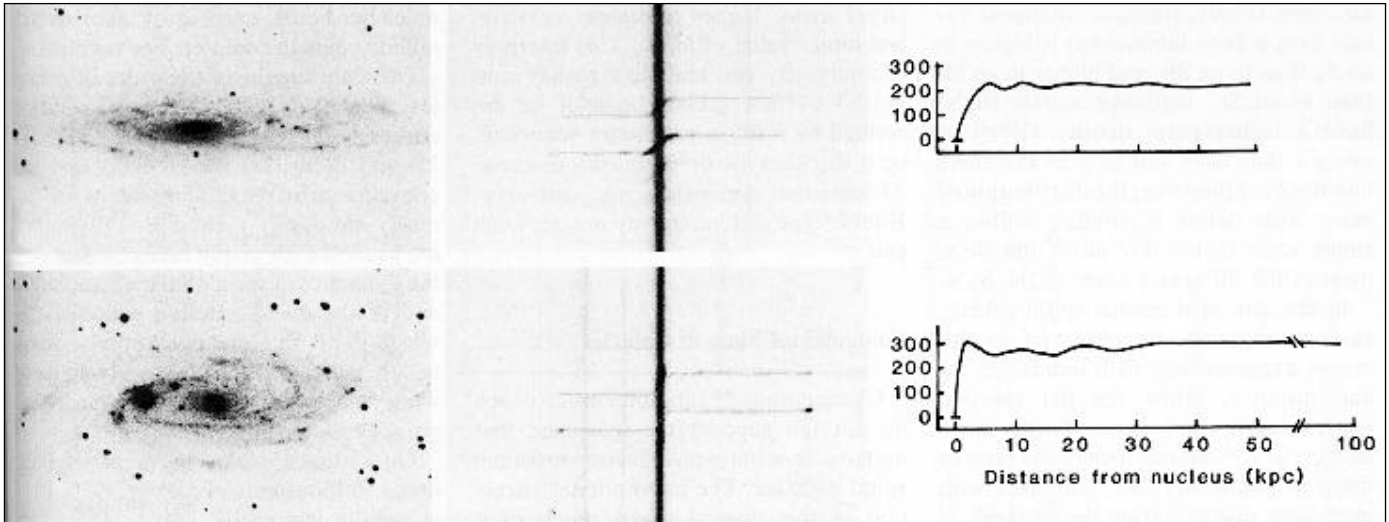


Figure 4. Evidence for the Existence of Dark Matter

The figure shows observations of two galaxies from which their rotations can be inferred; the inferred rotations provide evidence for the existence of dark matter. On the left are photographs of the galaxies NGC 801 and UGC 2885. The images in the center are photographs of their spectra made by allowing light from the galaxies to diffract through a horizontal slit. Therefore each photograph is the spectrum of a horizontal strip through the center of the galaxy; vertical position indicates wavelength. The heavy vertical lines in the center are the spectra of the galactic nuclei, which are highly luminous at all wavelengths. The step-shaped horizontal lines show the wavelengths of spectral lines of hydrogen and nitrogen as a function of distance from the center. Their shapes result from the Doppler blueshifts and redshifts due to the rotations of the galaxies; the wavelengths of these lines are shorter on the side of the galaxy rotating toward us and longer on the side rotating away from us. Thus the average orbital speeds of stars as a function of radius can be deduced from the displacements of those lines as a function of distance from the center of the galaxies. The resulting “rotation curves,” combining data from both sides of the galaxies, are on the right. (The vertical axis shows the rotational velocity in kilometers second⁻¹) Note that beyond a few kiloparsecs the velocity is nearly constant. This velocity distribution is typical of spiral galaxies. An entirely different velocity distribution, decreasing to zero as $r^{-1/2}$, would be expected if the masses of the galaxies were concentrated at their centers, as are their distributions of luminous matter. The only convincing explanation of the observed velocity distribution is that the galaxies are embedded in massive halos of unobservable “dark matter.” (From an article by Vera Rubin, The rotation of spiral galaxies, *Science* 220: 1339–1344, 24 June 1983. Copyright AAAS. The photographs of the galaxies were made by B. Carney using the 4-meter telescope at Kitt Peak National Observatory.)

Background”), about ten thousand years after the Big Bang, marks the time when, at least in principle, the force of mutual gravitation attraction could begin causing matter to collapse into the large-scale structures we see.

A later major event was recombination, when nuclei of hydrogen and helium combined with electrons to form stable neutral atoms. At that time the background radiation, consisting of photons in thermal equilibrium with the matter, decoupled from the neutral atoms and expanded freely to become

the cosmic background radiation now observed. As explained in the sidebar “Big Bang Cosmology and the Microwave Background,” any density perturbation on a distance scale λ present at the time of recombination, t_{recomb} , should be imprinted on the cosmic background as a temperature variation on a distance scale $\lambda/a(t_{\text{recomb}})$. Thus the cosmic background provides a direct measure of the matter distribution at the time of recombination and an observational constraint on models of large-scale structure formation.

So far the resolution of the instruments on the COBE satellite limits measurements of inhomogeneity to very large distance scales, too large to differentiate the various models of large-scale structure formation. Nevertheless, density inhomogeneity was seen and its amplitude was large enough to be consistent with the idea that gravity was decisively responsible for the formation of structure.

Initial conditions of the CDM model. The standard CDM model pos-

tulates very specific initial conditions for the development of large-scale structure. First it assumes that $\Omega = 1$, as suggested by inflationary models of the very early universe. Second, in line with the upper limit on the density of ordinary, or baryonic, matter allowed by primordial nucleosynthesis calculations (see “Big Bang Cosmology and the Microwave Background”), the standard CDM model assumes that 95 percent of the matter is nonbaryonic dark matter that is essentially noninteracting with ordinary matter. It also assumes that the dark matter was cold, or moving at nonrelativistic speeds, by the end of the radiation-dominated era. Finally, it postulates a scale-free spectrum of primordial density fluctuations—similar to the spectrum predicted by inflationary models. However, nonbaryonic cold dark matter has never been detected. So why should anyone take these assumptions about dark matter seriously?

Evidence for dark matter. The existence of dark matter (not necessarily cold) was first proposed by Fritz Zwicky in 1933 to explain how high-velocity galaxies observed in the very dense cluster known as the Coma cluster could remain gravitationally bound. More mass must exist in that cluster than was visible as luminous matter. In the 1970s a similar type of observation was made on the scale of single galaxies. Figure 4 shows the circular components of the orbital velocities of stars and gas clouds in two spiral galaxies deduced from very careful redshift measurements; the data are plotted as a function of radius r from the centers of the galaxies. In each galaxy the circular velocities, $v_{\text{circ}}(r)$, outside a radius of a few kiloparsecs are all approximately equal to a constant, v_{const} . Similar results are found in all spiral galaxies. On the other hand, quite a different prediction follows from the

concentration of luminous matter toward the centers of those galaxies. The centrifugal force on a star orbiting the galactic center at a radius r must equal the gravitational force, or

$$\frac{mv_{\text{circ}}^2}{r} = \frac{GmM(<r)}{r^2},$$

where m is the mass of the star and $M(<r)$ is the mass of the galaxy inside the radius r . If the mass of the galaxy is distributed similarly to the luminous matter, one would conclude that the velocities of stars in roughly circular orbits far from the center, $v_{\text{circ}}(r)$, are proportional to $r^{-1/2}$, as is the case for planets in our solar system. But in spiral galaxies (including our own—see Figure 4) this is never the case!

To resolve the contradiction, astrophysicists postulated the existence of invisible (dark) matter distributed such that $M(<r)$ increases approximately with the radius:

$$M(<r) \approx \frac{v_{\text{const}}^2 r}{G} \propto r.$$

This distribution is less concentrated toward the center of the galaxy than the distribution of luminous matter and provides the additional gravitational force needed to keep the stars in orbit at the observed velocities.

The dark matter in a typical bright galaxy would form an invisible halo, as shown in the opening spread of this article, and would have to be at least ten times more massive than the luminous matter to explain the observed motions of stars in the galaxy. The estimate of Ω would therefore increase from about 0.01 (deduced from luminous matter) to about 0.1. All that matter might be made of baryons and still not violate the constraints on baryon density pro-

vided by primordial nucleosynthesis. Indeed, recent observations of a phenomenon called gravitational microlensing suggest that dark matter in the form of old stars or dense planet-like objects, presumably made of baryonic matter, is present in galactic halos.

On scales of megaparsecs, the translational motions of galaxies, especially those in clusters of galaxies, seem to imply the existence of still more dark matter, enough to bring the value of Ω to 0.2 ± 0.1 . On the largest scales observed (30 to 100 megaparsecs), the densities deduced from comparison of the translational motions with the distribution of matter suggest that Ω must be at least as large as 0.3, and probably close to 1, which is the value assumed in the CDM model. Perhaps the most compelling argument in support of a large amount of cold dark matter is that its existence may well explain the observed structure of the universe with the fewest assumptions and the most natural physics.

The composition of cold dark matter. What might be the composition of cold dark matter? Particle theorists have helped attack this interesting problem by offering up a whole list of possible candidates. Those candidates, such as axions and photinos, have been predicted in theoretical models that extend the standard model of particle physics. They are called weakly interacting massive particles (WIMPs) and have both properties necessary to constituents of cold dark matter: First, they would interact very, very weakly with the rest of matter and radiation; in fact they would be practically undetectable. Second, they would have sufficiently large masses (typically, much larger than the mass of the proton) that their thermal velocities near the end of the radiation-dominated era would have been far slower than the speed of

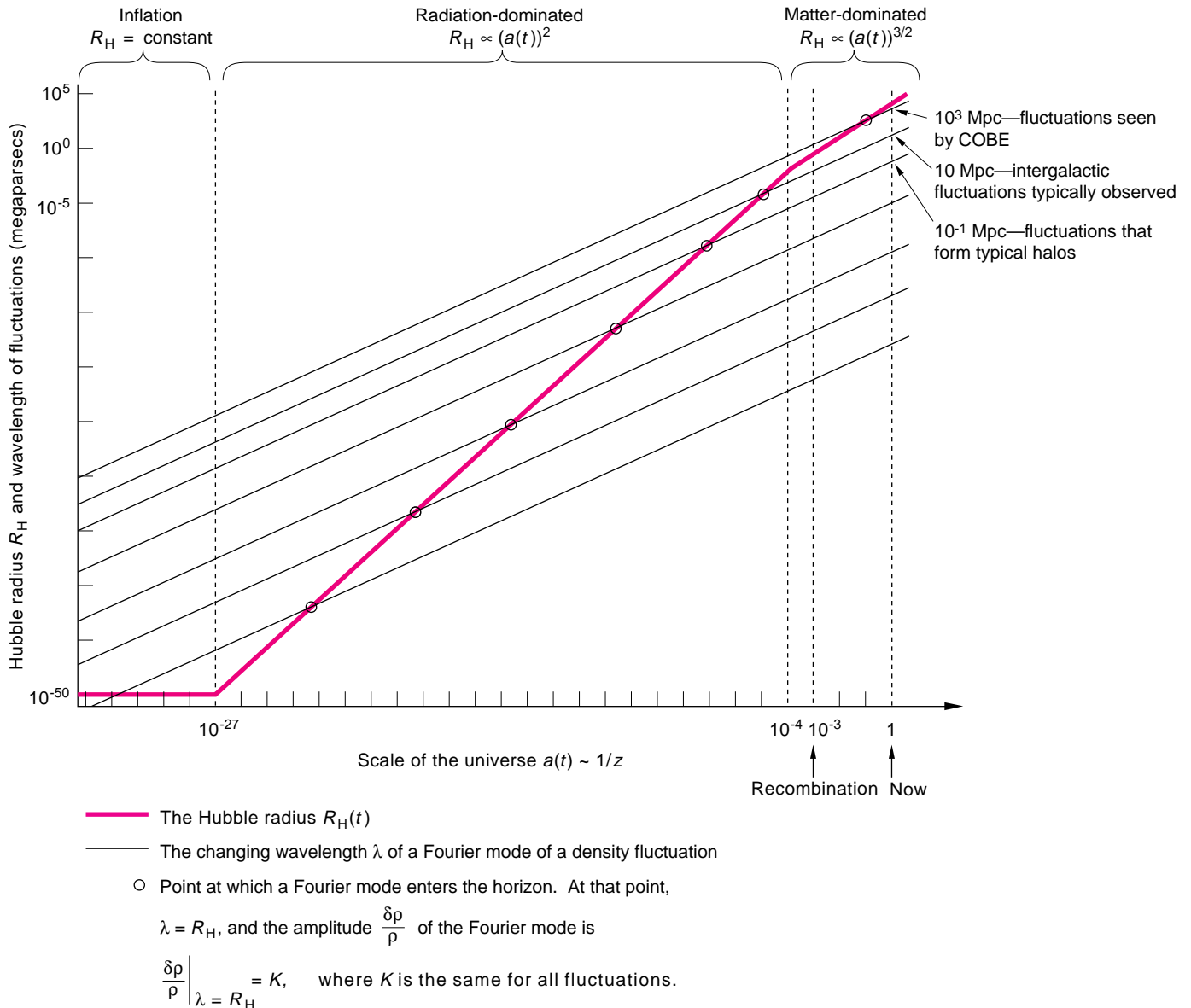


Figure 5. Horizon Crossing and the “Scale-Free” Spectrum

The log-log plot shows the wavelengths of the Fourier modes of the primordial density fluctuations, λ , as they stretch out because of the expansion of the universe. Here the expansion is measured by the scale factor, $a(t)$. Also shown is the Hubble radius, $R_H(t)$ (red), as a function of $a(t)$. As noted in the sidebar “Big Bang Cosmology and the Microwave Background,” $a(t)$ increases exponentially with time (as $\exp(H_{\text{inflation}} t)$) during inflation, as $t^{1/2}$ during the radiation-dominated era, and as $t^{2/3}$ during the matter-dominated era. Since the Hubble radius is defined as $R_H(t) \equiv c/H(t) \equiv ca(t)/(da/dt)$, it is equal to the constant $c/H_{\text{inflation}}$ during inflation; it increases as $(a(t))^2$ during the radiation-dominated era, and it increases as $(a(t))^{3/2}$ during the matter-dominated era. Therefore after inflation the wavelength of each mode increases more slowly than $R_H(t)$. The wavelength of each mode crosses the Hubble radius at a particular time (circled on the graph), depending on the initial wavelength of that mode. At that time the fluctuation is said to “enter the horizon.” The Harrison-Zel’dovich spectrum is defined to be scale-free in the sense that each mode has the same amplitude at the time it enters the horizon, or $\delta\rho/\rho \Big|_{\lambda = R_H(t)} = K$, where K is the same for all modes.

light.* Density fluctuations in cold dark matter on galactic and smaller scales would therefore not be wiped out by the free streaming of the cold-matter particles from more dense to less dense regions. Consequently, those fluctuations could begin to grow under the influence of gravity at the start of the matter-dominated era, as postulated by the CDM model.

Primordial fluctuations in cold dark matter. The power spectrum is a convenient and succinct way of characterizing density perturbations in many (but not all) cosmological models. Density fluctuations as a function of position \mathbf{r} can always be expressed in terms of a sum of elementary sinusoidal ripples, that is, through the Fourier expansion of $\delta\rho(\mathbf{r}) \equiv \rho(\mathbf{r}) - \bar{\rho}$, the deviation of the density from its average:

$$\delta\rho(\mathbf{r}) \propto \sum_{\mathbf{k}} a_{\mathbf{k}} \cos(\mathbf{k} \cdot \mathbf{r} + \phi_{\mathbf{k}}).$$

Above, \mathbf{k} are the wavevectors of the ripples, $a_{\mathbf{k}}$ are the amplitudes of different modes corresponding to different wavelengths $\lambda = 2\pi/|\mathbf{k}|$, and $\phi_{\mathbf{k}}$ are the phases. In a large class of cosmological models (including CDM), it is assumed that phases do not matter, that they are random and uncorrelated between different modes. When this is true, one can focus solely on the amplitudes $a_{\mathbf{k}}$ and characterize the fluctuations by their average values, that is, by the power spectrum:

$$P(k) = \langle a_{\mathbf{k}}^2 \rangle,$$

where $k \equiv |\mathbf{k}|$. In addition, the deviations of density from the average, $\delta\rho(\mathbf{r})$ at various points \mathbf{r} , turn out to have a Gaussian distribution.

Cold-dark-matter cosmology starts with the assumption that primordial

*Axions are an exception: They are less massive, but move slowly for other reasons.

density perturbations are Gaussian and their power spectrum has the form

$$P(k) \propto k,$$

which is known as the Harrison-Zel'dovich spectrum. This spectrum follows approximately from the scale invariance of the process of inflation. (Inflation is a postulated period of exponential expansion at very early times, which solves certain cosmological problems related to causality.) This scale-free spectrum was actually proposed before inflationary models on the basis of an elegant feature: In a universe with $P(k) \propto k$ and $\Omega = 1$, each density fluctuation has the same amplitude at the time it enters the horizon of the observable universe, that is, at the time its wavelength λ is equal to c times the age of the universe, or approximately equal to the Hubble radius, $c/H(t)$.

Figure 5 shows the increase in wavelength of each mode with time due to the expansion of the universe; note that the horizon increases faster than the wavelengths of the modes, so as time goes on, larger and larger modes fulfill the criterion that their wavelengths are about equal to $c/H(t)$, at which times they enter the observable universe. Such a universe has the appealing feature of being cosmologically scale-invariant: At any time the only important scale is defined by the size of the horizon at that time.

Linear growth of density fluctuations. The initial density fluctuations postulated by the CDM model would grow under the influence of gravity because, in a distribution of matter that is nearly uniform in density except for small “ripples,” the gravity of denser regions tends to attract more mass from nearby regions, so the ripples become larger. As they become larger, they become even more effective at attracting

matter, and so they continue to grow, possibly becoming progenitors of galaxies or galaxy clusters. In the CDM model and most models other than HDM, gravitational collapse occurred on small scales first and then on larger scales. Thus globular clusters formed before galaxies, and galaxies before clusters of galaxies.

Until the time of recombination and for a long time afterward, the growth of the density fluctuations can be modeled analytically because their average amplitude, or $\delta\rho/\bar{\rho}$, was small enough for the growth to be linear. Linear growth is calculated simply by computing the independent growth of each mode—the evolution of the power spectrum with time. In this linear regime the fate of each mode depends on whether it enters the horizon in the radiation-dominated or in the matter-dominated era.

Modes that enter the horizon in the radiation-dominated era are, in effect, ripples in the density of a plasma that is dominated by photons. The pressure of the plasma prevents gravitational development of these ripples and causes them to oscillate as sound waves. This state of affairs persists until t_{eq} , the time at which the energy density of matter equals the energy density of radiation. Only after t_{eq} , when the radiative contribution to the energy density becomes negligible, can density perturbations begin to grow. Thus, the growth of the modes with wavelengths smaller than the horizon at t_{eq} is delayed, and therefore stunted. As shown in Figure 5, these modes include the ones that develop into galaxies.

By contrast, the modes that enter the horizon in the matter-dominated era, well after t_{eq} , begin to grow immediately as a result of gravitational attraction. Thus by the time of recombination, when the fluctuations made a lasting imprint in the cosmic background, the shape of the power spectrum $P(k)$

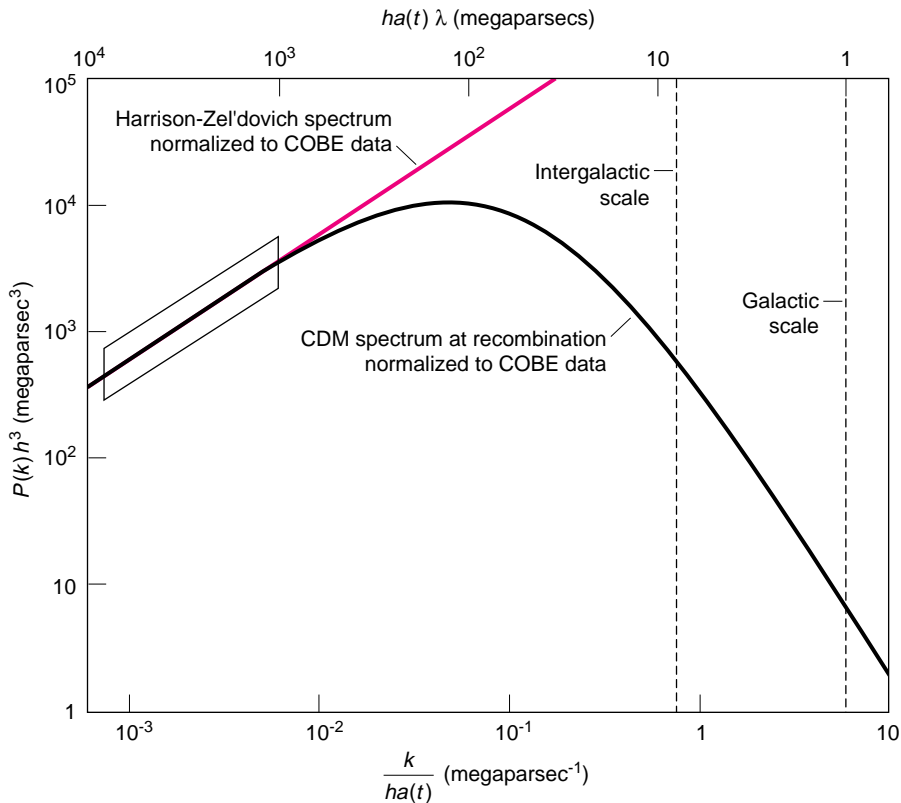


Figure 6. The Density-fluctuation Spectrum of Cold Dark Matter

Plotted is the density-fluctuation power spectrum, $P(k)$, of cold dark matter at the time of recombination. (For comparison with present structures, the horizontal scales give the wavenumbers k and wavelengths λ of the modes now, when the universe is $a(t_0)/a(t_{\text{recomb}}) \approx 1000$ times bigger.) The spectrum is normalized to agree with the COBE observations; the parallelogram at left shows the range allowed by the COBE data. Also plotted is the Harrison-Zel'dovich spectrum, $P(k) \propto k$ (gray), with the same normalization. As discussed in the text, the CDM spectrum has less power at small scales (large k) than the Harrison-Zel'dovich spectrum, because fluctuations on those scales enter the horizon during the radiation-dominated era, when they cannot grow. The relatively small scale labeled “intergalactic” is roughly the scale used for observations of the galaxy distribution (also shown in Figure 5). The comparison of the galaxy distribution to the CDM spectrum will be discussed in the text.

had changed from the primordial Harrison-Zel'dovich spectrum: Modes with wavelengths less than $c/H(t_{\text{eq}})$ had less power relative to modes with wavelengths greater than $c/H(t_{\text{eq}})$ than was the case in the primordial spectrum (see Figure 6).

Details of the CDM power spectrum depend somewhat on the assumed frac-

tion of the matter that consists of baryons because the baryonic density distribution is affected by interactions with radiation during the radiation-dominated era, but for reasonable values of Ω_{baryon} those effects are minor. The crucial assumption determining the CDM power spectrum is that the dark matter is composed of cold particles at

the start of the matter-dominated era—when the gravity of dark matter begins to matter!

Early computer simulations of nonlinear growth. Given the power spectrum of density fluctuations, $P(k)$, at the time of recombination, one can, in principle, compute the subsequent evolution of the matter distribution in the CDM universe for comparison with observations. The computation requires very large-scale computer simulations for two reasons. First, on scales of tens of megaparsecs and smaller, the observed amplitudes of density fluctuations $\delta\rho/\bar{\rho}$ are on the order of 1 and larger, and so their growth must have been nonlinear. Second, in order to resolve inhomogeneities on the relatively small scales of galaxies, 10 kiloparsecs, and at the same time show the distribution and motion of matter on the largest observed scales, many tens of megaparsecs and greater, the computation must involve a very large number of massive particles.

The early computer studies, initiated about ten years ago right after the CDM model was proposed, were inconclusive because they could not include enough particles. Typically, about ten thousand very massive particles represented all the matter in a region 10 to 100 megaparsecs across, and a single particle with a mass on the order of 10^{12} solar masses represented all the stars and dark matter in a galactic halo. Thus there was no convincing criterion for deciding which of those particles were prospective galactic halos that would become luminous due to star formation and which of those particles represented matter that would remain dark. Nevertheless those early “one-particle-per-galaxy” simulations gave hope that the CDM cosmology was right: The value of the normalization of the power spectrum could be chosen so that the

simulations reproduced the observed relative motions of galaxies on small intergalactic scales (on the order of 1 megaparsec). They also reproduced the observed spatial distribution of galaxies provided one introduced a bias, an ad hoc assumption that the nonuniformity of galaxies exaggerates the nonuniformity of dark matter by some factor b greater than 1. More formally, if we define the number of galaxies in a given volume as N and the total mass in that volume as M , then on average

$$\delta N/\bar{N} = b \delta M/\bar{M}.$$

Those early studies required a bias b of 2.5, or equivalently only particles in very dense regions (on scales of 1 megaparsec) were identified as prospective galaxies. Later, smaller values of the bias and correspondingly larger values for the normalization of the power spectrum had to be adopted to account for observations on scales around 50 megaparsecs such as the famous “Great Attractor”—an observed flow of galaxies, including ours—and the distribution of rich clusters of galaxies called Abell clusters.

The introduction of a bias to match simulations with the observed spatial distribution of galaxies is not unreasonable, because the spatial distribution of galaxies is determined not just by the distribution of total matter in the universe, which is modeled by simulations, but also by the highly uncertain process of star formation. The data that more directly reflect the underlying matter distribution, and that therefore must be matched by simulations, are the observed peculiar velocities of galaxies, motions that are due not to the expansion of the universe, but to gravitational forces on galaxies from inhomogeneities in the local mass distribution, including dark matter. Therefore peculiar velocities have long been considered a

better measure of mass inhomogeneity than correlations of galaxy positions.

Since galaxies are too far away for the detection of their motion transverse to our line of sight, only the components along our line of sight of their velocities—the components that affect redshift—are observed. Peculiar velocities show up as deviations from Hubble’s law—provided there is some way to determine the distance to galaxies independent of velocity, or redshift, measurements. The naive method is to assume that galaxies near each other in the sky are members of the same cluster; then differences in their redshifts would arise from differences in their peculiar velocities rather than from differences in distance.* The actual procedure for determining peculiar velocities is a sophisticated statistical application of the same principle. The analysis yields the component along the line between two galaxies of the inferred peculiar-velocity difference between them. That component is called the pairwise radial velocity. The distribution of peculiar velocities is characterized by the standard deviation of pairwise radial velocities, which is written as σ_v . Measurements seemed to indicate that for pairs of galaxies on the order of 1 megaparsec apart, σ_v is 300 to 400 kilometers per second.

Before the 1992 COBE results, scientists would vary the normalization of the power spectrum to achieve agreement between the peculiar velocities predicted by the simulation and those observed and then vary the bias to match the observed spatial distribution

*These differences in radial velocities are responsible for the distorted appearance of clusters in galaxy maps based on redshift, such as Figure 1a. The Coma Berenices cluster in that figure probably has an approximately spherical shape, but the peculiar velocities of its galaxies give them widely different redshifts, elongating the plotted spatial distribution of the cluster into a “finger of God” pointing directly at Earth.

of luminous matter. Since none of the simulations were able to achieve high resolution on both large and small scales simultaneously, the results were ambiguous.

Cold Dark Matter, Large Scales, and COBE

The freedom to vary the normalization of the power spectrum and “fine-tune” the bias disappeared in early 1992 after the announcement that the COBE satellite had detected microwave-background fluctuations of a few parts in 10^6 on scales of 1000 megaparsecs and higher. Since the shape of the CDM fluctuation spectrum is determined, the COBE measurements fix the normalization constant and thus the entire spectrum.*

The COBE results also determine the bias b between the distribution of matter and the distribution of luminous matter. One extrapolates the CDM spectrum to the present and compares the resulting spectrum to the present amplitude of density fluctuations. The result for scales around $10/h$ megaparsecs (the intergalactic scales shown on Figures 5 and 6) is $\delta\rho/\rho \approx 1$ with a standard deviation of about 20 percent. One can compare that predicted amplitude with the quantity conventionally used to report the distribution of luminous matter, namely, the amplitude $\delta\rho/\rho$ on the scale of $8/h$ megaparsecs, known as σ_8 . That is,

$$\sigma_8 \equiv \left. \frac{\delta\rho}{\rho} \right|_{\lambda = 8/h \text{ Mpc}}.$$

*There is still some freedom in the baryonic content of the universe and, perhaps more important, in the exponent of the primordial power spectrum of density fluctuations, which would be exactly scale-invariant only if it were generated by endless inflation.

The observed value of σ_8 for luminous matter is also about 1; thus the COBE data exclude the possibility of a significant bias.

High-resolution state-of-the-art simulations. Spurred by the COBE observations, we recently carried out large computer simulations of structure formation in a CDM universe. The simulations were large enough to resolve the formation of prospective galaxies on kiloparsec scales and to model the structure and motions of sheets and filaments of matter on scales four orders of magnitude larger.

The simulations involve either 9 or 17 million point particles moving under the influence of mutual gravitational attraction in an expanding spherical fragment of the universe. (Since the particles have no internal degrees of freedom, and the only interaction in the simulation is gravity, no energy is dissipated.) The fragment expands during the simulation to a final diameter of 100 or 250 megaparsecs at a redshift of $z = 0$, which corresponds to the present time, $t = t_0$. For numerical reasons the simulation starts at a redshift z between 50 and 100, that is, at a time well after the time of recombination ($z = 1000$) but before the onset of nonlinear growth. To determine the density-perturbation spectrum at the start of the simulation, we use linear theory to calculate the spectrum as a function of z from the CDM spectrum at recombination under the assumption that the fluctuations grow linearly with the scale factor. The universe is assumed to be flat, the Hubble parameter is assumed to be 50 (km/s)/Mpc, and the normalization of the spectrum in each of the runs is selected to bracket the value suggested by the COBE measurements. We cannot do this directly, but rather fine-tune the normalization to yield the desired value of σ_8 at $z = 0$.

The spectrum of fluctuations at the starting value of z is built into the simulation by arranging the particles in a regular array and assigning them different masses selected at random to match the predetermined spectrum. To include the effects of the universal expansion, the particles are given initial velocities in accordance with Hubble's law. The simulation then follows the motion of this collection of particles. At each timestep, the changes in the position and velocity of each particle are found by first calculating the gravitational force on each particle produced by the other particles and then integrating Newton's second law of motion over the duration of each timestep. The forces are calculated according to the familiar formula for Newtonian gravity.

The use of Newtonian gravity together with initial conditions in accordance with Hubble's law may appear to be a poor man's version of the general model for a relativistic universe, but it can be shown (see the caption of Figure 3) that, to a good approximation, a region of the universe really does behave according to that description provided that the region is matter-dominated and is significantly smaller than the Hubble radius c/H_0 . Nevertheless, only after general relativity was understood did scientists become bold enough to apply Newton's laws to the cosmos.

In a standard simulation of N particles, the time needed to calculate the gravitational force between each pair of particles and to sum up the total force on each particle in each timestep is proportional to N^2 . When N is in the millions, as in our simulations, that time is prohibitively long even on the most powerful computers. We have been able to reduce the time significantly by replacing pairwise interactions between distant particles with the well-known multipole approximation for the gravitational force exerted by a group of parti-

cles on a distant particle. The approximation is implemented by using a hierarchical algorithm called a treecode in which the problem domain is subdivided into cubes of decreasing size, the first terms of the multipole expansion are computed for the particles in each cube, and the multipole approximation of the gravitational force is applied whenever the accompanying errors are negligible. Treecodes calculate forces involving millions of particles thousands of times faster than conventional algorithms. Their execution time depends on the number of particles as $N \log N$ rather than N^2 , a very significant reduction when N is large.

Our treecode and its potential applications in other fields are described in "A Fast Tree Code for Many-Body Problems." The simulations presented here were run on the Intel Touchstone Delta, a parallel supercomputer owned partly by the Laboratory and installed at Caltech.

Figure 7 illustrates the development of structure in the simulation. A typical final distribution of particles appears in Figure 8. Many of the clumps shown in those figures are made up of hundreds of particles and have masses in the range characteristic of a galactic halo. We interpret those clumps as prospective galactic halos; the final distribution of halos is shown in Figure 9, and a detailed distribution of both particles and halos in a small area is shown in Figure 10.

Once the halos are identified, we analyze local halo dynamics including internal rotation, collapse, and merger. At the same time, since each simulation generates approximately 10,000 such halos, we can work with statistically significant numbers in investigating the relative motions and overall spatial distribution of halos on scales of tens of megaparsecs. The size of the halos and their spatial distribution are not in any

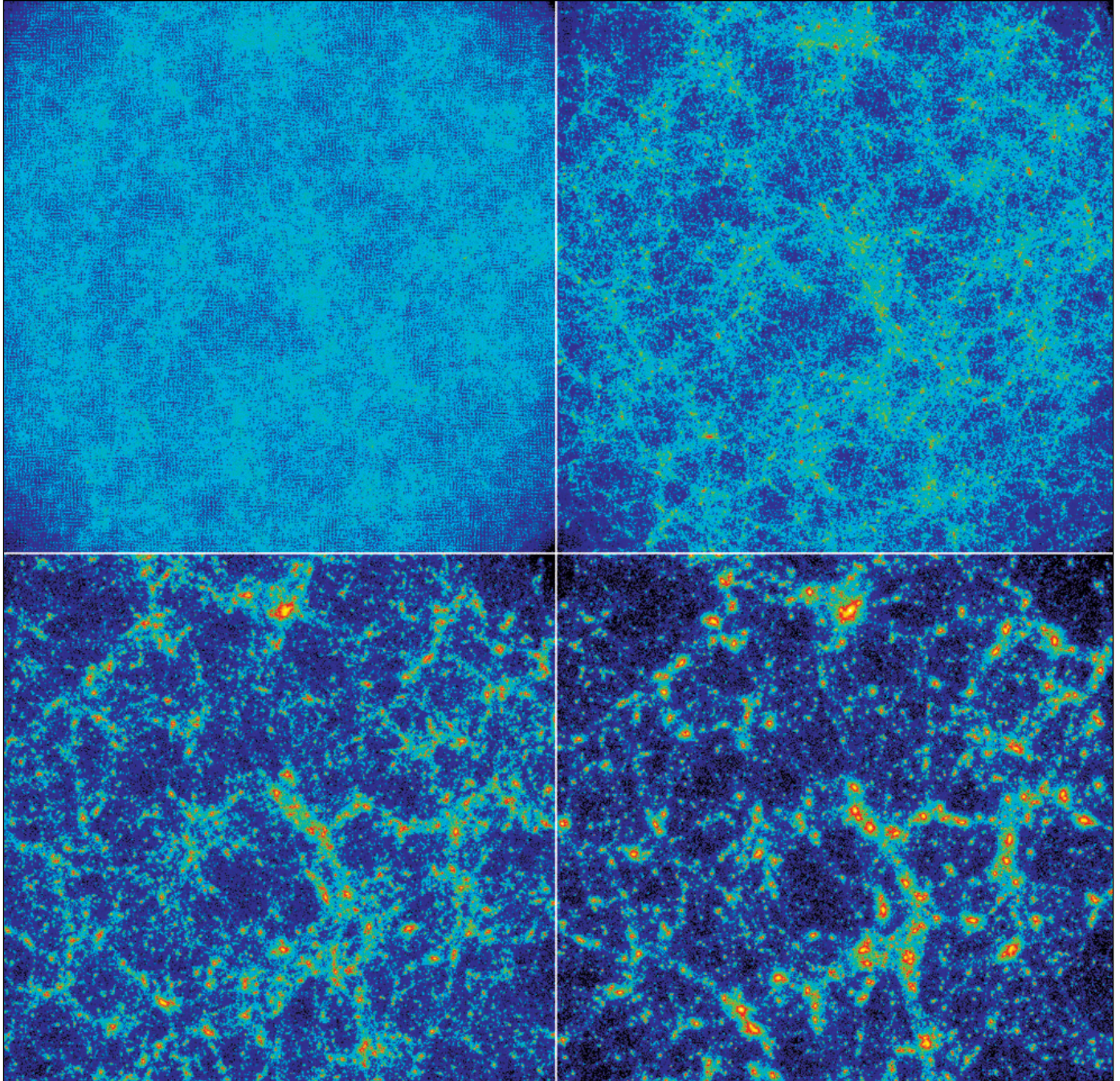


Figure 7. The Evolution of Structure in a Simulation

These four frames show consecutive stages of the particle distribution in one of our simulations of the matter distribution. The simulation starts a few tens of million years after the Big Bang ($z = 64$), shortly before density fluctuations began to grow nonlinearly. The first frame is at $z = 10$; the last is at the present. The color of each pixel indicates the logarithm of the particle density along the line of sight through the computational volume; blue indicates the lowest density, then cyan, green, red, yellow, and white. As the simulation progresses, the mass becomes more clumped, eventually forming structures qualitatively similar to those observed. The region of space shown expands as the universe expands; it is about 200 megaparsecs across in the final frame.

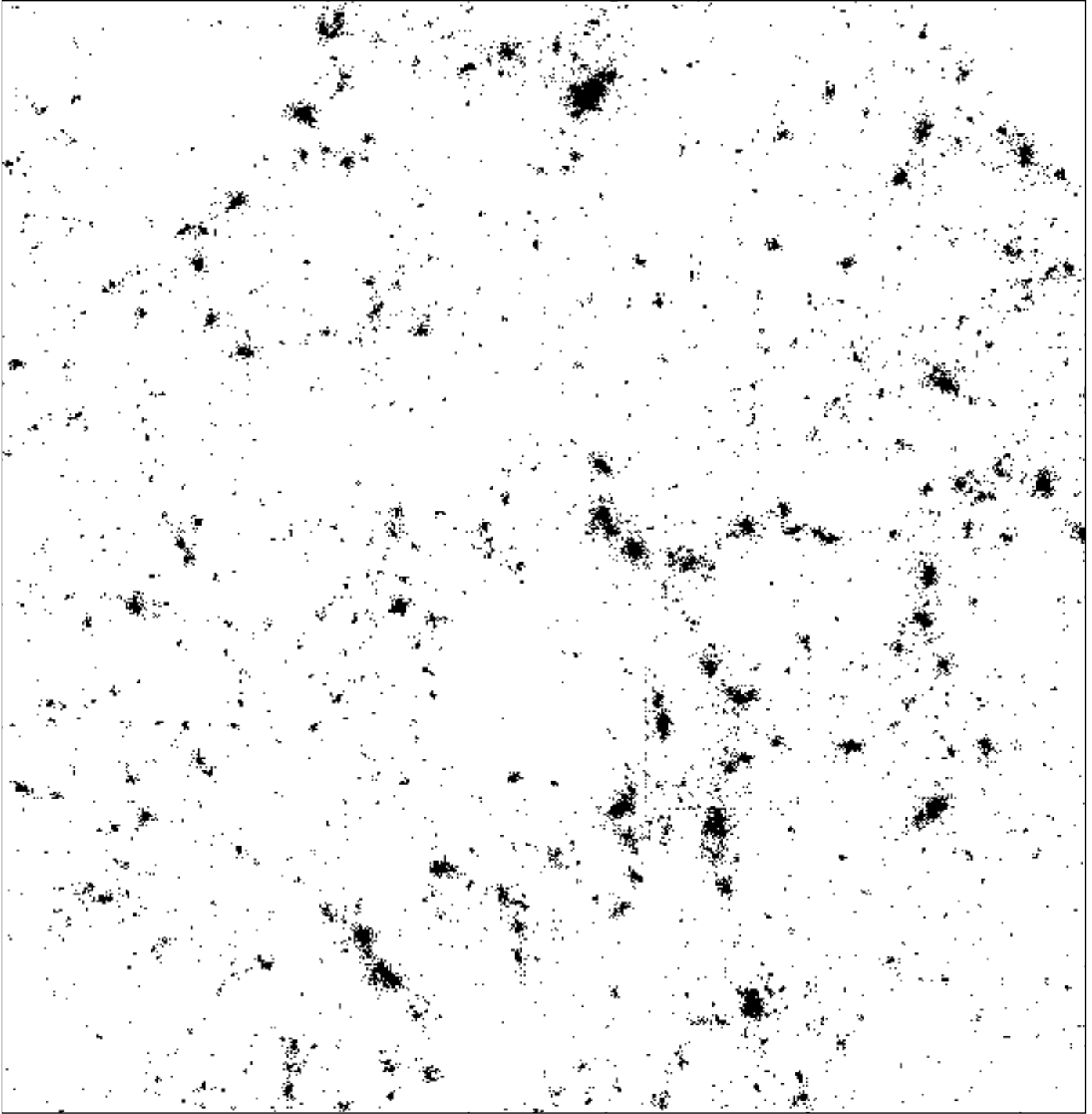


Figure 8. Particles at the End of a Simulation

The figure shows the projected location of particles in a sphere of diameter 250 megaparsecs at the end of a simulation. The distance across the image is about 180 Mpc (part of the sphere is cut off). Only about one in every thirty particles is plotted. The figure shows that the distribution is inhomogeneous on all scales. The square shows the region enlarged in Figure 10. The treecode used for the simulation is described in "A Fast Tree Code for Many-Body Problems," accompanying this article.

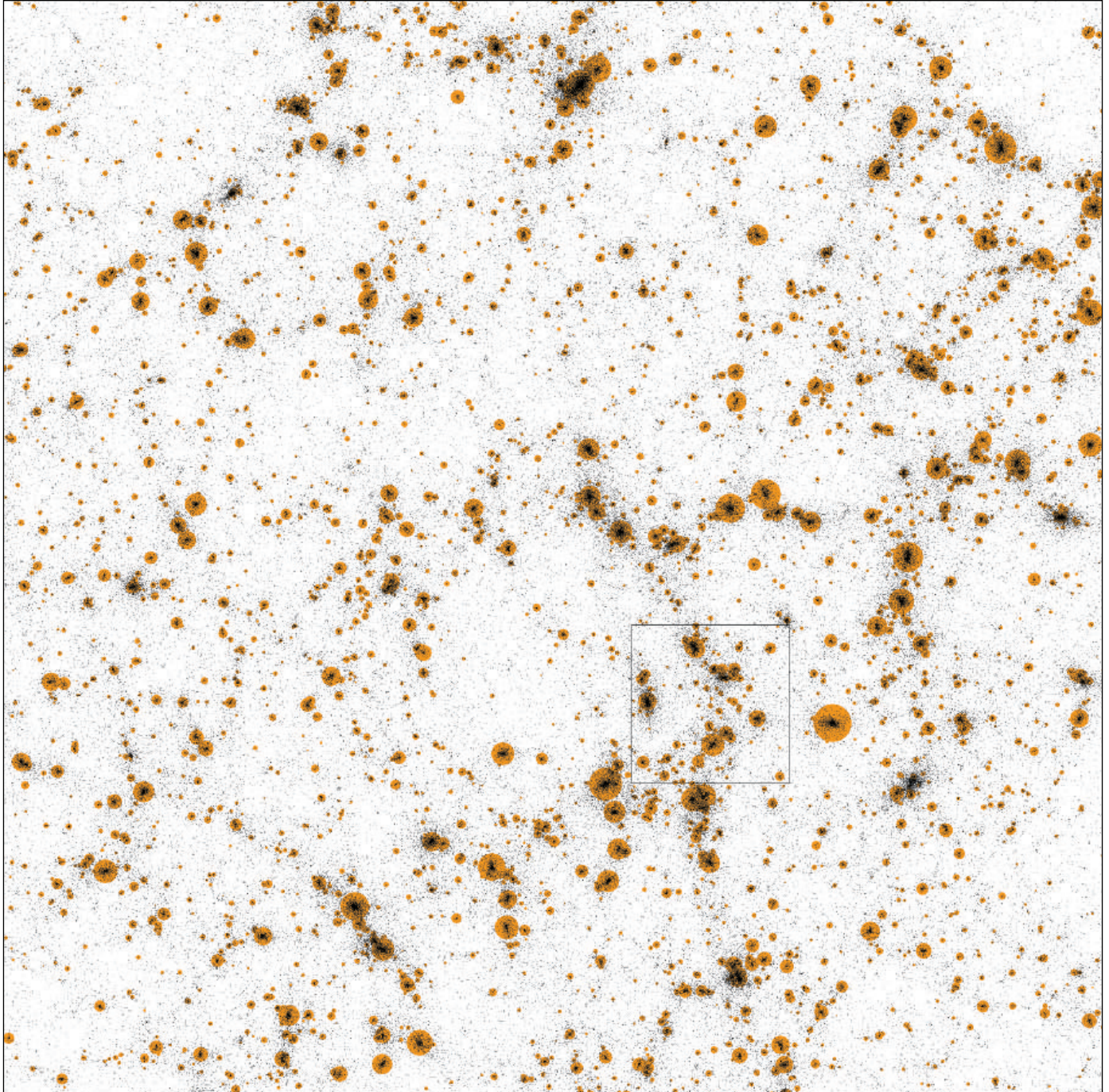


Figure 9. Halos at the End of a Simulation

A plot of the location of “galactic halos” in the system of particles shown in Figure 7, as determined by an algorithm that locates clumps of particles. Halos are defined to be regions whose centers are at least 10,000 times denser than the average density of the universe and contain at least 10 particles. The sizes of the circles are proportional to the masses of the halos. There are about 6000 such halos in this picture. Note that clusters, voids, and sheets (which appear as narrow concentrations on this two-dimensional projection) are present, as in the observations.

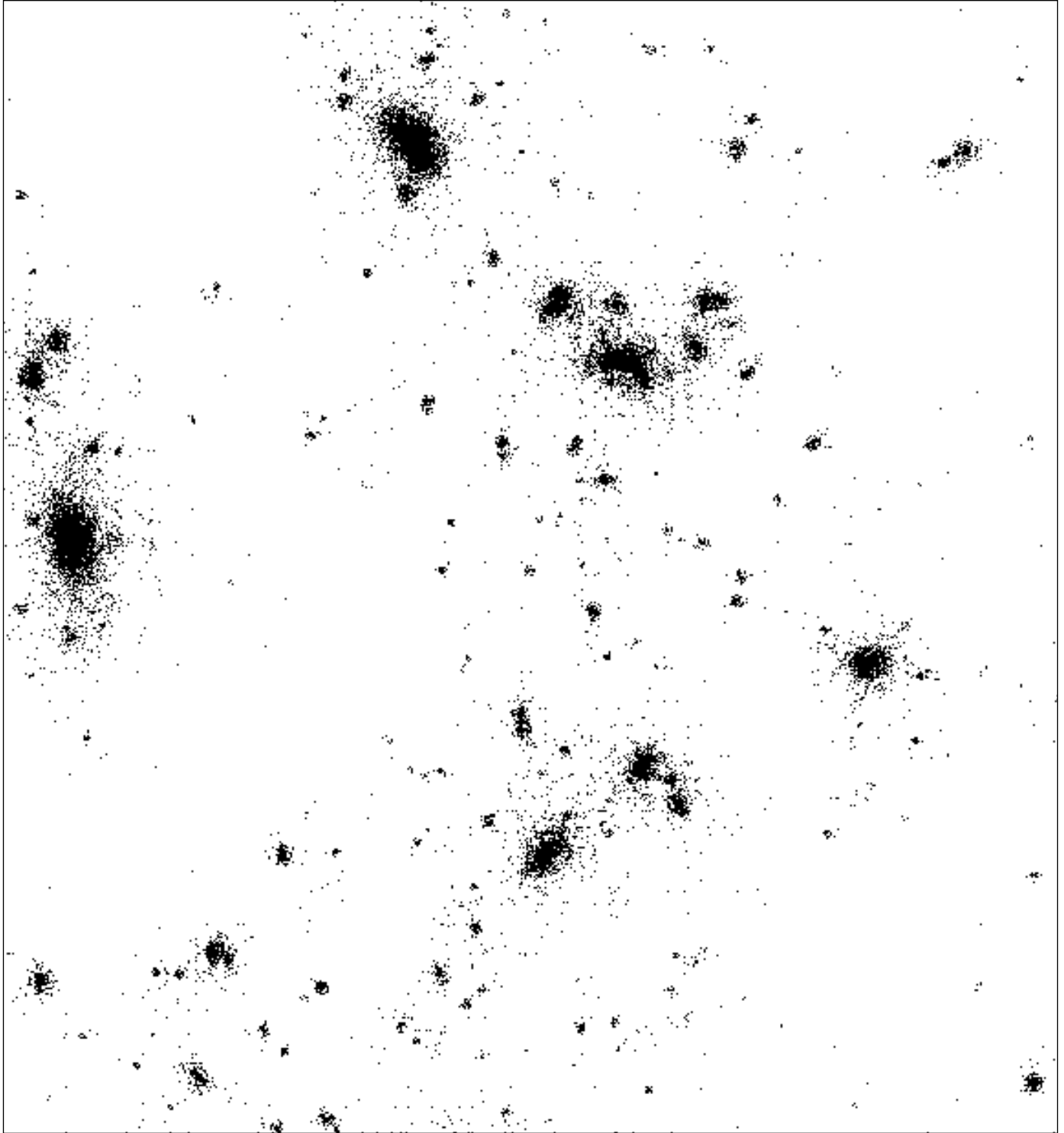


Figure 10. Detailed Particle Distribution at the End of a Simulation

A high-resolution plot of every particle (about 250,000) in one fiftieth of the total system (indicated by the small square in Figure 8). The circles represent galactic halos as identified in Figure 9.

obvious disagreement with observation, as can be shown by computing halo-halo correlation functions as well as other measures of the halo distribution, and comparing them with similar quantities computed on the basis of observational data. To evaluate the distribution of mass within each halo and check whether it is compatible with the observations, we used the positions of the tens or hundreds of particles in each halo to determine for each halo $M(<r)$, the mass within a radius r as a function of r . Then, using the previously discussed formula for circular velocity, $v_{\text{circ}} = \sqrt{GM(<r)}/r$, we determined the orbital velocities that stars would have around the center of the halo. The mass distributions of our halos do indeed look like those inferred from the observations of spiral galaxies: $M(<r)$ in our simulated halos is roughly proportional to r at large r , so orbital velocities of stars would be roughly independent of distance (see Figure 4). The average orbital velocity, or equivalently the average mass, of the halos also seems to be in general agreement with observation. Figure 11 is a plot of the number of prospective galactic halos versus the inferred orbital velocity of a star at a distance of 100 kiloparsecs from the center of the halo. The two curves in the figure are derived from simulations with different normalizations of the density-fluctuation spectrum, one specified by the COBE value of σ_8 and the other by a somewhat lower value. The peaks in both curves, at a velocity of 150 kilometers per second, are probably an artifact of our resolution—we cannot resolve halos less massive than about one tenth of the mass of our galaxy. A more important feature is that relatively few orbital velocities are much higher than the orbital velocity of stars in the Milky Way around the galactic center, about 260 kilometers per second. In previous

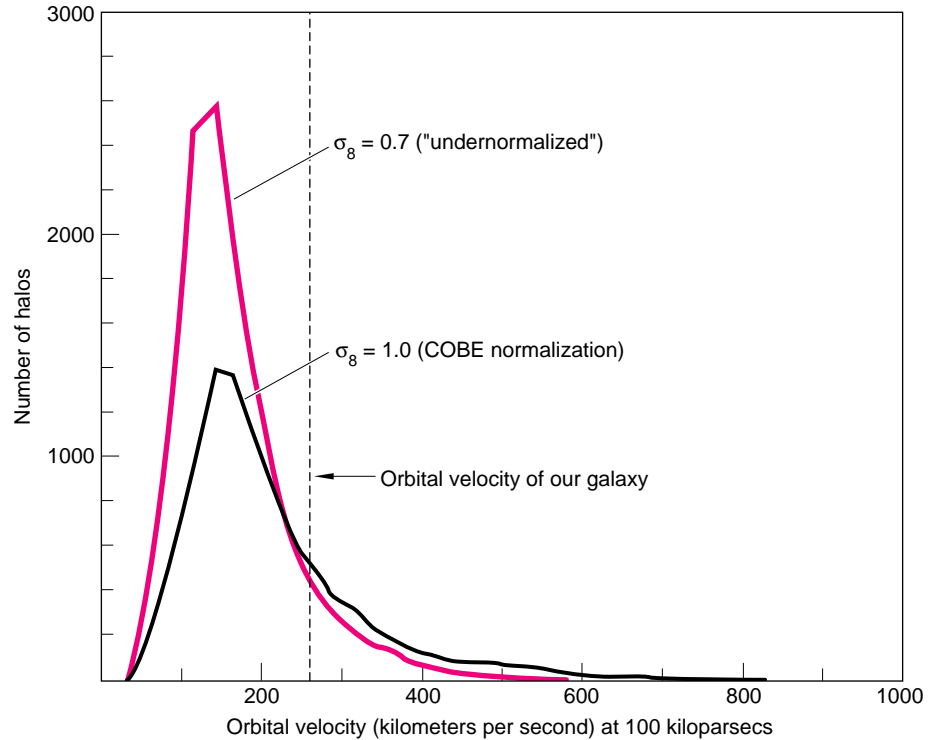


Figure 11. Distribution of Halo Number as a Function of Inferred Orbital Velocities of Stars in Halos

The graph shows the number of halos in our simulation versus the inferred orbital velocity, v_{circ} , of a star around the center of each halo at a radius of 100 kiloparsecs. The radius of 100 kiloparsecs was chosen because orbital velocities of stars around galactic centers are usually independent of distance at such large distances (as in Figure 4). The velocity associated with each halo is inferred from the mass distribution within the halo. The most abundant halos produced in our simulation are those with the lowest masses that the simulation can resolve, and thus they usually have low orbital velocities. Therefore the peaks of the curves shown are artifacts of the simulation, not characteristics to be compared with observations. We show results from runs of the simulation using two different normalizations of the density-fluctuation spectrum: $\sigma_8 \approx 1.0$ (black curves), the naive prediction from COBE measurements, and a lower normalization, $\sigma_8 \approx 0.7$ (red curves), which, as argued in the text, may be a much better estimate. The vertical line at 260 kilometers per second indicates the velocity of stars orbiting the center of our galaxy; the highest velocities observed in other galaxies are about 600 kilometers per second. Thus the rotational velocities predicted by our simulation are on the right order of magnitude, indicating that the local mass distribution within halos is about right. The lower normalization gives better agreement with observations since it does not lead to orbital velocities in excess of 600 kilometers per second.

simulations, low resolution had caused halos to merge into overly large halos, so the inferred orbital velocities were considerably higher than those observed.

Apart from that “overmerging” problem, CDM cosmology never had much difficulty accounting for spatial structure on megaparsec scales and smaller.

The model ran into serious trouble only when it had to account simultaneously for both the spatial distribution and relative motions of galactic halos. Figure 12 shows results from our simulations for the root-mean-square pairwise radial velocities of halos as a function of the distance separating the halos. (The root-mean-square pairwise radial veloc-

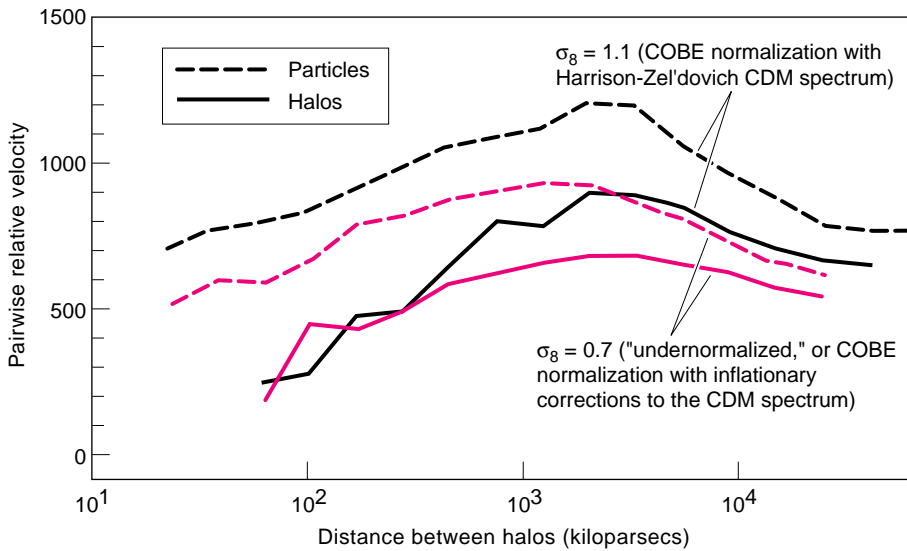


Figure 12. Pairwise Radial Velocities of Halos

The solid lines indicate the root-mean-square value of the pairwise radial velocity—the differences of the velocities of two halos along the line connecting them, excluding the Hubble expansion—as a function of separation of the halos. The dashed lines show the relative velocities of particles in the simulation rather than those of halos. As in Figure 11, the simulation was run twice, using the COBE normalization (black curves) and a lower normalization (red curves). The data from the lower-normalization run are in the same range as the observed pairwise radial velocities of real galaxies, provided the same statistical procedures are applied to the simulated and observed data (see discussion in the main text).

ity is closely related to σ_v , the standard deviation of pairwise radial velocities. The difference in our simulation is about 20 percent.) Again the two sets of results derive from the different normalizations used in the simulations. For the lower normalization the σ_v of halos separated by 1 to 2 megaparsecs is on the order of 600 kilometers per second. Figure 13 shows the distribution of pairwise radial velocities for halos separated by 1 to 2 megaparsecs.

The value of σ_v of 600 kilometers per second obtained from our high-resolution simulation is less than half of the value indicated by the old “one particle per galaxy” CDM simulations, but still appears to be too large compared with the usually quoted observational value— 340 ± 40 kilometers per second on megaparsec scales. At first we thought that this discrepancy ruled out the CDM scenario. However, having re-examined observed and simulated relative velocities more carefully, we have concluded that the quoted value for σ_v is not a very reliable diagnostic

for evaluating the global dynamics of simulations. In particular, the values deduced can vary by nearly an order of magnitude from dense clusters to underdense “backwaters” like our own local group of galaxies. Similarly, large variations in σ_v (factors of 2 to 5) are seen among fragments of our simulated universes containing roughly 1400 galaxies. These fragments are comparable in size to the regions of the universe usually surveyed to measure σ_v .

The usually quoted value of σ_v was obtained by M. Davis and P. J. E. Peebles from their analysis of the sample of about 1200 galaxies in the northern sky compiled by the Harvard-Smithsonian Center for Astrophysics. This sample contains the galaxies in the Virgo cluster, which move at high speeds and tend to affect relative velocities to an extent disproportionate to their small numbers. In addition, overall infall into the Virgo cluster results in significant infall velocities quite far from its core. Davis and Peebles employed a special procedure to compensate for some of

those effects. When we reanalyzed the observations without such special treatment, we found that the standard deviation of the pairwise radial velocities was around 500 kilometers per second, in close agreement with our simulations, especially those with the lower normalizations. Thus, contrary to recent popular prejudice, the observations of pairwise radial velocities do not seem to rule out CDM cosmology.

Our results for the relative velocities of halos and the orbital velocities within halos agree particularly well with observations when we take the value of σ_8 to be 0.7, rather than 1.0 to 1.3 as has been inferred from the COBE measurements of temperature variations. We would argue that the agreement is not pure happenstance, but rather that the value close to 0.7 is the relevant value for galaxy formation. First, the density-fluctuation power spectrum resulting from inflation is proportional not to k but to $k^{1-\epsilon}$, where $\epsilon = 0.02$ to 0.05. This small difference is not important at the long wavelengths (small k) measured by COBE, but it does affect the extrapolation of the COBE results to the shorter-wavelength (larger k) modes relevant to galaxy formation, as it reduces the predicted amplitude of modes on megaparsec scales. Second, COBE infers differences in temperature from differences in radiation intensity, but those differences can be caused by gravity waves as well as by density variations. Gravity waves do not develop into astronomical structures. Therefore the part of the COBE temperature-fluctuation data due to gravity waves must be estimated and subtracted to obtain the part due to density fluctuations.

These two effects are interrelated and for small ϵ have similar magnitudes. Taken together, they imply that the primordial density perturbations at short wavelengths (large values of k) have less power than that usually in-

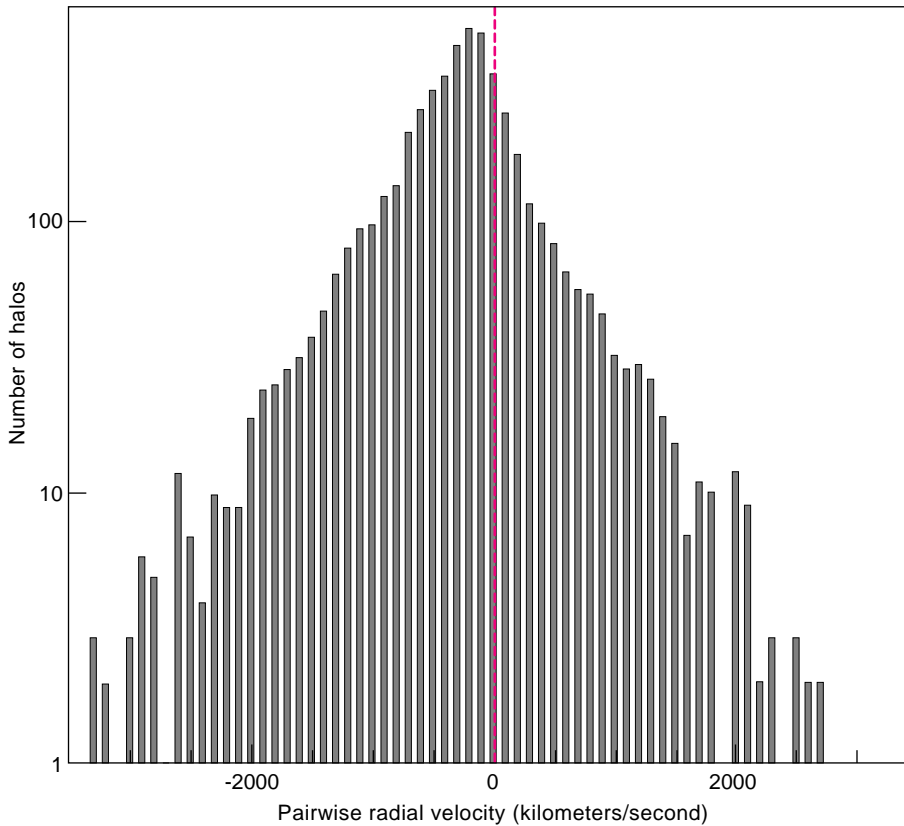


Figure 13. Distribution of Pairwise Relative Velocities of Halos at 1–2 Megaparsec Separations

The graph shows the number of pairs of halos in which the two halos are separated by 1–2 megaparsecs versus the pairwise relative velocities of the two halos in the pair. These results are from a simulation with $\sigma_8 \approx 0.7$. The center of the distribution is not at 0 but at -280 km/s; the negative average velocity indicates a net infall, or movement of halos toward each other, that is, toward local mass concentrations. This infall, due to gravitational attraction, is similar to observed movements of real galaxies. The local increases in density as nearby halos move toward each other are examples of the gravitational growth of inhomogeneity. Note that the graph appears as a triangle on this log plot, so the distribution of velocities is exponential. Therefore σ_v at this separation is simply proportional to the slopes of the sides of the triangle.

ferred from the COBE data; extrapolating that result to the scale of our simulations implies that σ_8 is about 20 to 40 percent less than the COBE data would seem to imply. Thus the agreement between our “undernormalized” simulations and the observational data might not be fortuitous; rather it might be telling us that inflation really did happen and that the resulting primordial fluctuation spectrum was only approximately of the Harrison-Zel’dovich form. If so (and admittedly we are getting carried away with optimism here), our simulations would provide the first concrete example in which detailed effects of inflation on the primordial fluctuation spectrum have to be taken into account to reconcile theoretical predictions with observations!

Nevertheless, the CDM model is still not “out of hot water.” Although the discrepancy of relative velocities of halos was considered the most serious difficulty with the model, there are other tests that deal more directly with the spatial distribution of galaxies. One observational uncertainty that will have a major impact on assessing the validity of cosmological models is the present value of the Hubble parameter, H_0 . All of the models that assume that $\Omega = 1$ (as does the standard CDM model) would be seriously endangered if H_0 turned out to be significantly larger than the currently favored value, 50 (km/s)/Mpc. The value of H_0 (in combination with the assumption that the universe is “flat,” that is, that $\Omega = 1$) fixes the age of the universe. According to present ideas of stellar evolution, the oldest stars we observe are approximately 14 billion years old, which is about equal to the age of the universe if $H_0 = 50$ (km/s)/Mpc and $\Omega = 1$. If H_0 turns out to be 100 (km/s)/Mpc, those stars would be twice as old as the universe! Such a discrepancy would cause, of course, a major “paradigm

shift” in cosmology, as it could be accommodated only by admitting that we are missing some important aspects of stellar evolution (which appears unlikely) or that the universe is open—has $\Omega \approx 0.1$.

Conclusions

The general framework of Big Bang cosmology has been very much strengthened by the COBE results. The cosmic background radiation appears to have a spectrum (that of emission from a perfect black body) and a temperature consistent with the predictions of the Big Bang theory. Furthermore, the COBE observations showed that the form of the primordial perturbations of the density of the universe was consistent with an approximately scale-free spectrum and that their amplitude was sufficiently large that they could act as seeds for the gravitational development of the currently observed density fluctuations, so that no forces other than gravity need to be invoked. The traditional problems with the CDM scenario follow from the *excess* of power on small scales and the resulting excess of the peculiar velocities between the galactic halos (and, therefore, presumably, between galaxies). We believe, however, that the results of our simulations are consistent with the observations when the simulations and observations are analyzed in the same way. Therefore there is no reason yet to abandon the CDM scenario.

Nevertheless, there is an almost embarrassing richness of insufficiently explored alternatives, one of the more popular being hybrid models involving both hot and cold dark matter. Fortunately, all of the alternatives will soon confront even more precise observational data, and their success will be evaluated by means of much more accurate

simulations. Indeed, the list of models ruled out by COBE and other observations (with the help of computer simulations) is becoming longer than the list of viable models. “Experimental cosmology,” using parallel supercomputers and sophisticated software, will undoubtedly help bring about continued progress in this exciting and fundamental field. The computer is becoming as important as the telescope in shaping our understanding of the universe. ■

Further Reading

- G. Efstathiou, J. R. Bond, and S. D. M. White. 1992. COBE background radiation anisotropies and large-scale structure in the Universe. *Monthly Notices of the Royal Astronomical Society* 258: 1P–6P.
- Margaret J. Geller and John P. Huchra. 1989. Mapping the universe. *Science* 246: 897–903.
- P. J. E. Peebles. 1993. *Principles of Physical Cosmology*. Princeton University Press.
- Vera C. Rubin. 1983. The rotation of spiral galaxies. *Science* 220: 1339–1344.
- Frank H. Shu. 1982. *The Physical Universe: An Introduction to Astronomy*. University Science Books.
- G. F. Smoot *et al.* 1992. Structure in the COBE differential microwave radiometer first-year maps. *The Astrophysical Journal* 396: L1–L5.
- M. S. Warren, P. J. Quinn, J. K. Salmon, and W. H. Zurek. 1992. Dark halos formed via dissipationless collapse. *Astrophysical Journal* 399: 405–425.
- W. H. Zurek, P. J. Quinn, J. K. Salmon, and M. S. Warren. 1993. Large-scale structure after COBE. Los Alamos National Laboratory unclassified release LA-UR-93-2104. *Astrophysical Journal*, in press.
- W. H. Zurek, P. J. Quinn, J. K. Salmon, and M. S. Warren. 1993. The second coming of cold dark matter? Los Alamos National Laboratory unclassified release LA-UR-93-4240. In *Proceedings of the Ninth IAP Meeting*, edited by F. Bouchet, in press.



Michael S. Warren (left) received his B.S. in physics, engineering, and applied science from the California Institute of Technology and his M.S. and Ph.D. in physics from the University of California, Santa Barbara. He came to the Laboratory in 1990 as a graduate research assistant in the Theoretical Astrophysics Group and is currently performing post-doctoral work in the Fluid Dynamics Group. In 1992 Warren received the Gordon Bell Prize and the Intel Grand Challenge Computing Award for the work described in “A Fast Tree Code for Many-Body Problems.”

Wojciech H. Zurek was born in Poland and received an M.Sc. from the Technical University of Kraków. He received his Ph.D. in Physics from the University of Texas at Austin. Zurek joined the Laboratory’s Theoretical Astrophysics Group as an Oppenheimer Fellow in 1984 and has remained in that group ever since; he is now its leader. He is also an external faculty member of the Santa Fe Institute and an associate of the Cosmology Section of the Canadian Institute for Advanced Research. His research interests include cosmology, relativistic astrophysics, the quantum theory of measurements, the foundations of statistical physics, and the physics of information. In 1988 Zurek was awarded the Los Alamos Fellows Prize for his research on galaxy formation.

Big Bang Cosmology and the Microwave Background

Salman Habib and Raymond J. Laflamme

Big Bang cosmology implies a certain predictable course for the thermal history of the universe. Here we review that history, emphasizing the two classic pieces of evidence that support it: the relative abundances of the light elements produced by primordial nucleosynthesis and the existence, spectrum, and fantastic isotropy of the cosmic microwave background. We will also discuss the modifications that might be caused by the presence of cold dark matter.

Figure 1 outlines the calendar of events, or major epochs, in the history. It can be divided into two main periods: the radiation-dominated period, which lasted for approximately ten thousand years (10^{11} seconds), and the matter-dominated period from ten thousand years after the Big Bang to the present. During the first fraction of a second after the Big Bang, the constituents of the universe undoubtedly included the particles of the standard model of particle physics: leptons, quarks, and the gauge bosons that mediate interactions among them, all moving at velocities so close to the velocity of light that from the point of view of thermodynamics, they behaved as particles of radiation

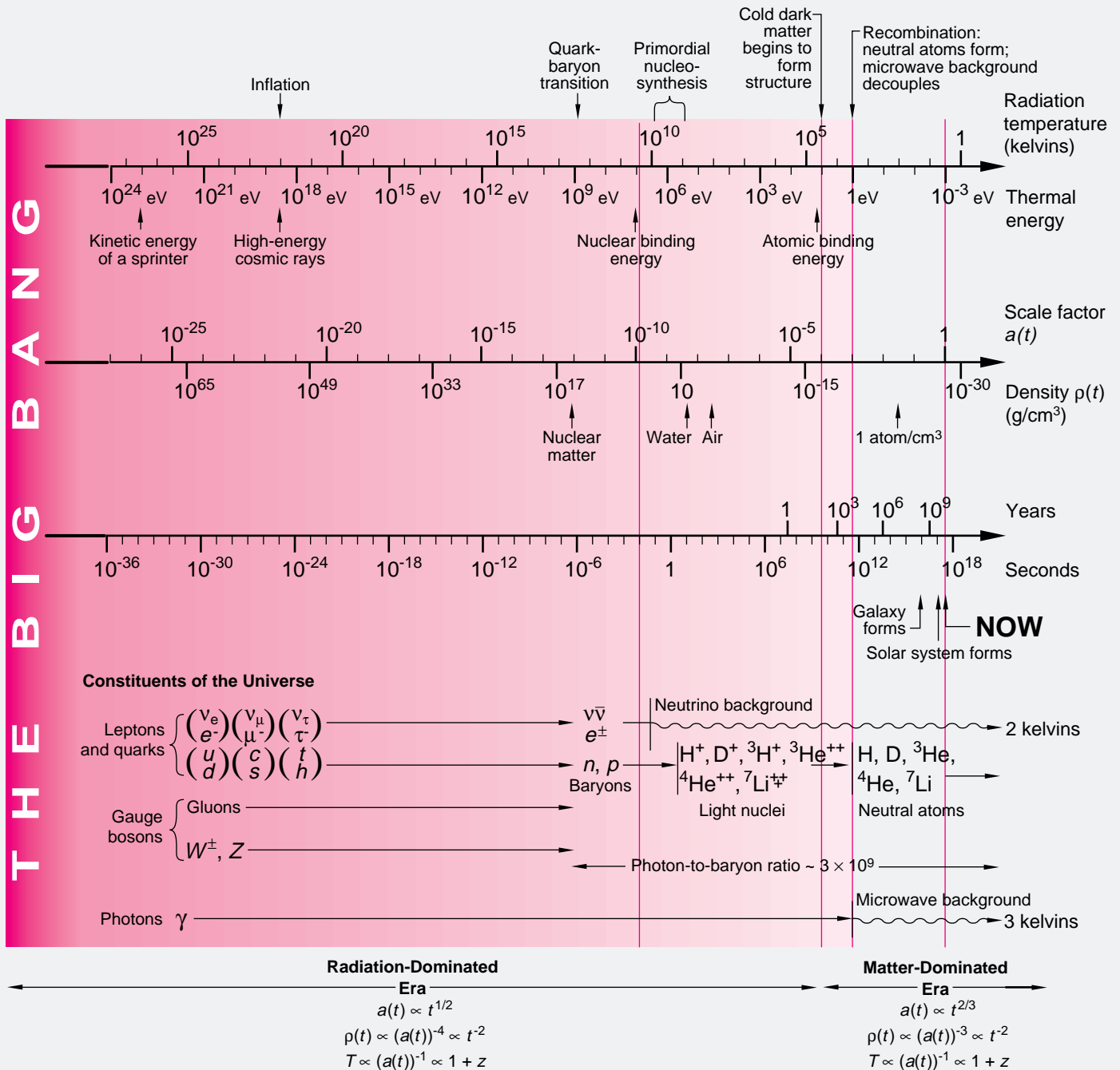
(that is, massless particles). The physics during this time is highly speculative. It is assumed that thermal energies were initially at the Planck scale (10^{28} eV), where the exotic and poorly understood phenomenon of quantum gravity could have played a crucial role. Many particle physicists and astrophysicists believe that a bit later, as the temperature cooled to 10^{26} kelvins, there was a brief period of inflation. During the inflationary phase, the size of the universe grew exponentially by a factor of at least 10^{28} . Less speculative is the prediction that at a temperature around 10^{12} kelvins, a quark-hadron transition occurred, when all the quarks and gluons combined to form protons and neutrons. The particles of cold dark matter, if they exist, would also have been created sometime after the inflationary phase.

The radiation-dominated era.

From about 10^{-2} second to the present, standard Big Bang cosmology presents an almost universally accepted picture, which is borne out by the COBE data and other recent observations. By that time the universe consisted of particles that were stable (or long-lived com-

Figure 1. The Thermal History of the Universe

The figure (adapted courtesy of Mike Turner) shows important epochs in the history of the universe according to standard Big Bang cosmology. It also shows the particles of matter and radiation constituting the universe during each epoch. The history is laid out on logarithmic scales to emphasize the early part. The scales are temperature, thermal energy kT , universal scale factor $a(t)$ (note that $a(t) = 1/(z + 1)$ where z is the redshift), density $\rho(t)$, and time. Temperature, thermal energy, redshift, and energy density (or the equivalent mass density) all decrease with time, whereas the scale of the universe $a(t) \equiv R(t)/R(t_0)$ increases. The temperature plotted is the temperature of the radiation, which until recombination is the temperature of the baryonic matter (neutrons and protons) as well. The formulae relating these variables are shown beneath the figure. They follow from conservation of energy and from the assumptions that matter and radiation were initially in thermal equilibrium and that the universe has expanded adiabatically since the hot Big Bang. The relationship between $\rho(t)$ and $a(t)$ changes depending on whether the universe is radiation-dominated or matter-dominated, that is, on whether most of the energy in the universe is in the form of radiation (photons, neutrinos, and other particles moving at speeds near c) or most of the energy is in the form of matter. The temperature and scale-factor axes are lined up with respect to each other by the observation that at present ($t = t_0$) the temperature of the cosmic background radiation is approximately 3 kelvins and that by definition $a(t_0) \equiv 1$. The time and density axes are lined up with the size axis by using a specific value for the Hubble constant and the assumption that the dimensionless density parameter Ω is equal to 1. Then one can place any event on all three axes if one knows its position on



one axis (usually temperature, sometimes redshift). Note that the scale-factor axis does not show the expansion due to inflation. In the lower portion of the figure, the line for neutrinos (and antineutrinos) turns into a wiggly line to indicate that at that time the neutrinos decoupled from matter and photons and subsequently expanded freely. The decoupling occurred when the universe became cool enough and dilute enough that neutrinos no longer interacted significantly with matter, at a temperature around 10^{10} kelvins. Similarly, at a temperature of about 3000 kelvins, the line for photons turns into a wiggly line to indicate their decoupling from matter due to recombination, the combining of the electrons with the positively charged nuclei to form neutral atoms. The subsequent free expansion of photons produced the cosmic microwave background radiation now observed.

pared to the age of the universe then): electrons, neutrinos, protons and neutrons, and photons, as well as the yet-unidentified constituents of dark matter. Unstable particles such as mu and tau leptons, mesons, and the gauge bosons of the weak force had disappeared through decay to stable particles. Figure 1 outlines the changes through time of the constituents of matter.

At about $t = 10^{-2}$ second, the universe was a “primeval fireball” consisting of baryonic matter and radiation in thermal equilibrium (and perhaps dark matter). Moreover, most of the energy was in the form of radiation. That follows from simply extrapolating the present density of matter and radiation back through the presumed adiabatic expansion of the universe. The energy density therefore obeyed the formula for black-body radiation, that is, for radiation in thermal equilibrium with a totally absorbing body. Thus the energy density u of each relativistic species present—whether photons, massless neutrinos, or even electrons and positrons moving at nearly the speed of light—obeyed the Stefan-Boltzmann law:

$$u_{\text{radiation}} = \bar{\rho}_{\text{radiation}} c^2 = \sigma T^4,$$

where T is temperature and σ is the analogue of the Stefan-Boltzmann constant.

Black-body radiation (even with a small admixture of equilibrated matter) behaves as a dissipationless fluid, so the radiation-dominated universe expanded adiabatically, and the total entropy was conserved. As mentioned in the main article in the discussion of the redshift, the adiabatic expansion “stretched” the wavelengths of photons and all other massless (or effectively massless) species in proportion to the universal scale factor $a(t)$, or $\lambda(t) = \lambda_0 a(t)$. Since all wavelengths were stretched

proportionally, the radiation continued to have a black-body spectrum, or Planck distribution, a fact that explains why the black-body spectrum of the cosmic background radiation is not a surprise. However, the temperature of the radiation, which is proportional to the average energy per quantum or inversely proportional to the average wavelength, decreased inversely with the scale factor, $T \propto (a(t))^{-1}$, or in proportion to the redshift ($T \propto z$). The energy density of the radiation therefore decreased due to both the stretching of the wavelengths of the quanta and the dilution of their number density; thus $\bar{\rho}_{\text{radiation}}$ decreased rapidly, as $(a(t))^{-4}$.

Equation 3 in the caption of Figure 3 in the main text is solved to relate the scale factor to time. In particular, when radiation was the major contributor to the energy density, $a(t)$ increased as $t^{1/2}$, and so the temperature of the universe decreased as $t^{-1/2}$. Thus during the radiation-dominated era, every decrease by a factor of 10 in temperature corresponds to an increase by a factor of 100 in time since the Big Bang. As the primeval fireball of matter and radiation expanded and cooled in thermal equilibrium, the decrease in temperature was like a cosmic clock, ticking off stages in the physics of the universe.

Primordial synthesis of the light elements. In the first few minutes after the Big Bang, the one major event that directly left observable traces was the primordial synthesis of the light elements through nuclear fusion reactions. In particular most of the helium-4 observed today in stars was almost assuredly synthesized then.

As the temperature fell to 10^9 kelvins ($t \approx 10^2$ seconds), conditions were right for nucleosynthesis to begin. Two quantities controlled the relative abundances of light elements that resulted. The first is the fraction of the

baryons that were neutrons, a fraction that decreased with time as neutrons were converted to protons through the weak interactions. Calculations predict that the number of neutrons at the time of primordial nucleosynthesis is approximately 12 percent of the total number of baryons and that almost all those neutrons ended up in helium-4 nuclei, the most tightly bound of the light-element nuclei. In other words, Big Bang cosmology predicts that the cosmic abundance by weight, or mass fraction, of helium-4 nuclei relative to hydrogen nuclei (protons) is about 24 percent, in agreement with observation.

The second determinant of relative abundances produced by primordial nucleosynthesis is the ratio of the number of photons to the number of baryons (or equivalently the photon entropy per baryon). That ratio does not change with time and therefore, as we will see below, places a constraint on the baryon density today. It also controlled the rates of various competing nuclear reactions at the time of nucleosynthesis. For example, nucleosynthesis could not really begin until the density of energetic photons became low enough that deuterons (deuterium nuclei) formed by the fusion of neutrons and protons (neutron + proton \rightarrow deuteron + photon) were unlikely to be blasted apart by the reverse reaction. That condition depends on the temperature, which must be about 10^9 kelvins, but also depends strongly on the ratio of photons to baryons. Once the deuteron abundance increased, nuclear reactions involving one or two deuterons could build on one another to form primordial abundances of all isotopes of hydrogen and helium: deuterium, tritium, helium-3, and helium-4 as well as very small amounts of lithium-7.

Primordial nucleosynthesis manufactured all those nuclei in abundances consistent with current data provided

the photon-to-baryon ratio was between 2.5×10^9 and 3.6×10^9 . The agreement between nucleosynthesis calculations and observations is considered convincing evidence that in the beginning the universe was hot, radiation-dominated, and in thermal equilibrium—that the hot Big Bang did indeed occur!

Further, since the ratio of photons to baryons during primordial nucleosynthesis was preserved to the present, it can be combined with the known number density of photons in the cosmic microwave background to predict the baryon density in the universe today. The bounds on the ratio yield bounds on the baryon density of $(2.5 \pm 0.5) \times 10^{-31}$ gram/centimeter³, and bounds on the baryonic contribution to Ω of $0.01 h^{-2} < \Omega_{\text{baryon}} < 0.015 h^{-2}$. If we assume that $h = 1/2$, then $\Omega_{\text{baryon}} \approx 0.05$, which is much larger than Ω_{luminous} . This prediction for Ω_{baryon} places a constraint on all models of structure formation.

The transition to a matter-dominated universe. Following nucleosynthesis, the cosmic soup consisted of photons in thermal equilibrium with hydrogen and helium nuclei and a number of electrons, left over from electron-positron annihilation, sufficient to balance the charge of the baryonic matter. The radiation-dominated universe continued to expand adiabatically, and entropy conservation continued to ensure that the temperature decreased inversely with the size of the universe, $T \propto (a(t))^{-1}$. Thus $\bar{\rho}_{\text{radiation}}$, the equivalent mass density of the radiation, continued to decrease as $(a(t))^{-4}$, whereas $\bar{\rho}_{\text{matter}}$ which was dominated by the matter's rest mass rather than its kinetic energy, decreased inversely as the volume, or as $(a(t))^{-3}$.

The matter-dominated era. Since the energy density of the radiation de-

creased more rapidly than that of the matter, at some point the energy density of the matter became larger, and from then on the universe has been matter-dominated. The temperature (or time, t_{eq}) at which the switch from a radiation-dominated to a matter-dominated universe occurred depends on the total matter density in the universe—baryons plus any species of dark matter that might be out there. If baryons make up all the matter and the total matter density today is 2.5×10^{-31} gram/centimeter³ (that is, $\Omega_{\text{baryon}} = 0.05$), then the switchover occurred at a temperature of about 1000 kelvins ($t_{\text{eq}} \approx 10^6$ years), whereas if $\Omega = 1$ and baryons make up only 5 percent of the matter, as assumed by the standard CDM model, then the universe became matter-dominated when the cosmic scale was a factor of 20 smaller or the temperature was a factor of 20 higher, at about 20,000 kelvins ($t_{\text{eq}} \approx 10^4$ years). Figure 1 was constructed with the assumption that $\Omega = 1$.

The beginning of the matter-dominated era marks the time when the mutual gravitational attraction of the matter became stronger than the gravitational pull of the background sea of radiation. Consequently matter could, in principle, begin to clump together under the influence of gravity. However, the baryons (primarily in the form of hydrogen and helium nuclei) and the electrons are electrically charged and therefore remained coupled to the photons through electromagnetic interactions, primarily the scattering of electrons and photons (Thomson scattering). Thus, their clumping was impeded by radiation pressure that was much stronger than gravitational forces in the nearly uniform mass distribution—the photon-to-baryon ratio is at least a billion to one—so density inhomogeneities could not grow in baryonic matter.

In contrast, non-baryonic dark mat-

ter, which by definition does not participate in electromagnetic interactions, began to develop density inhomogeneities when the universe became matter-dominated, provided the dark-matter temperature, or average thermal energy, was low enough and the masses of the dark matter particles were high enough that the particles were moving at nonrelativistic speeds. (Otherwise, small-scale fluctuations would have been washed out by the free streaming of dark-matter particles.) That is, the dark matter must have been “cold” for structure to have begun to form on all scales at t_{eq} , when the energy in matter was equal to the energy in radiation.

Recombination. When the universe cooled to about 3000 kelvins, the situation changed dramatically. That temperature is far enough below the ionization temperature of helium and hydrogen atoms that almost all the electrons combine with the helium and hydrogen nuclei to form neutral atoms. Cosmologists call that event “recombination” (a somewhat misleading term since the electrons and nuclei were combining for the first time). After recombination, charged particles were rare, and so the scattering of photons by matter was also rare. Thus the time of recombination, t_{recomb} , marks the end of the tight coupling between baryonic matter and photons; thermal equilibrium between matter and radiation is no longer maintained, and the universe is said to be transparent to radiation since photons can travel through the universe with little chance of being scattered.

Once the radiation pressure ceased to have an effect on baryonic matter, that matter too could begin to feel the pull of gravity. In fact, if dark matter were not present, recombination would have to mark the beginning of the growth of density fluctuations due to

gravitational forces. However, as stated in the introduction to the main article, recombination apparently occurred too late (and the density of baryonic matter was too low) for the growth of gravitational instabilities to explain the observed large-scale structure. Thus the majority of cosmologists believe dark matter must have led that development.

The discovery of the cosmic background radiation. According to Big Bang cosmology, the radiation that existed at the time of recombination was thermal and consequently had a black-body spectrum at a temperature of about 3000 kelvins; in other words, its energy density as a function of wavelength had the Planck distribution. At recombination, the universe became transparent to radiation, so the photons survived unchanged except for their redshift due to cosmic expansion. As the wavelength of each photon increased in proportion to the scale factor $a(t)$, the form of the spectrum was preserved; that is, the radiation now has a black-body spectrum but at a temperature lower by a factor $a(t_0)/a(t_{\text{recomb}})$ than the temperature at the time of recombination. That spectrum is shown in Figure 2. Indeed, Big Bang cosmology predicts that low-temperature black-body radiation should now be observable from all empty parts of the sky—this remnant of recombination forms a cosmic background.

The existence of a cosmic remnant left over from the hot Big Bang was first predicted by George Gamow, R. A. Alpher, and R. C. Herman in 1948 on the basis of their analysis of the conditions needed for primordial nucleosynthesis. In the early sixties R. H. Dicke and P. J. E. Peebles took this notion seriously enough to plan experiments to detect its presence. Ironically, in 1965, while Dicke, Peebles, D. T. Wilkinson, and P. G. Roll were building their

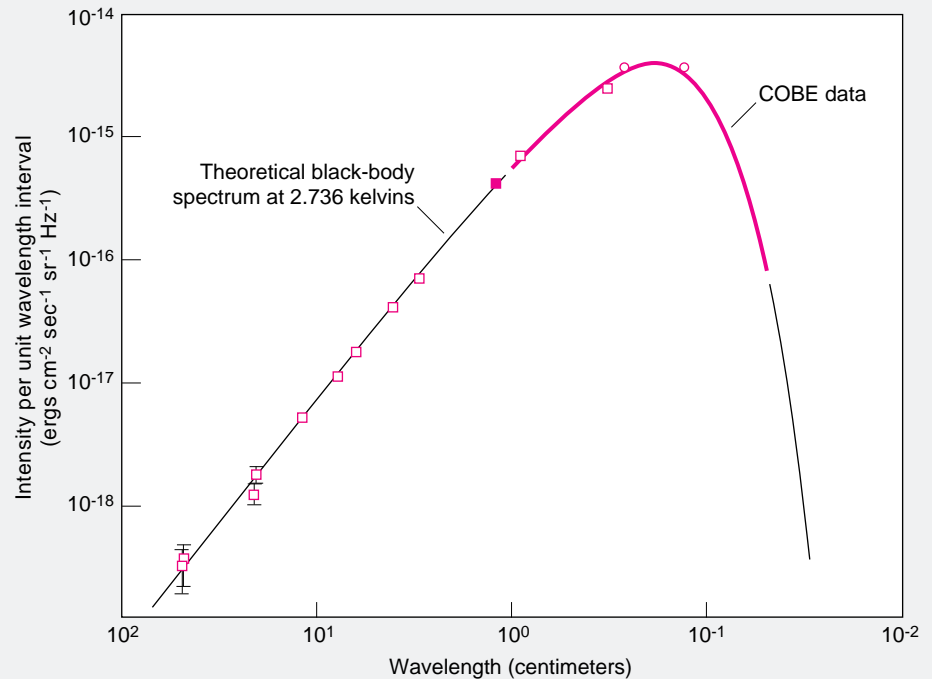


Figure 2. The Planck Distribution and the Cosmic Background Radiation The figure shows the Planck distribution, or the energy density per unit wavelength of radiation in thermal equilibrium with a perfectly absorbing (black) body, at a temperature of 2.736 kelvins. The red data points represent measurements by instruments other than COBE of the energy density per unit wavelength for the cosmic background radiation. The precision of most of the data points is so high that error bars would be no bigger than the symbols representing the points. The red part of the curve represents the measurements made by the COBE satellite; its error bars and deviations from the Planck distribution are too small to appear. Thus the background radiation appears to have a nearly perfect “black-body” spectrum, indicating that matter and radiation were in thermal equilibrium at the time of recombination, as predicted by standard Big Bang cosmology. (Figure adapted by courtesy of D. T. Wilkinson.)

equipment, the cosmic background radiation was detected inadvertently by Arno Penzias and Robert W. Wilson at Bell Laboratories. Penzias and Wilson were investigating the internal noise properties of an antenna designed to detect radio waves emitted by our own galaxy. Quite unexpectedly they discovered extra noise at a wavelength of 7.35 centimeters whose intensity was independent of direction. The intensity also did not vary with the time of day or the time of year. If that radiation was assumed to be one point of a

black-body spectrum, then the inferred temperature of the spectrum would be between 2.5 and 4.5 kelvins. Peebles, Dicke, Wilkinson, and Roll in a separate paper suggested that this highly unusual radiation was the predicted remnant of the Big Bang.

Since 1965 astronomers have measured the intensity of the cosmic background radiation at many wavelengths. In 1989 the COBE satellite was launched specifically to study various aspects of the radiation. The COBRA rocket-based experiments and one of

the instruments on COBE confirmed to very high precision that the spectrum of the cosmic background is indeed a black-body spectrum. COBE made a particularly precise measurement of the temperature: 2.736 kelvins plus or minus a few millikelvins. A black-body spectrum at that temperature, shown in Figure 2, has a peak at the wavelength of 2 millimeters, which lies in the microwave region of the electromagnetic spectrum. The ratio of the present temperature of the cosmic microwave background to the photon temperature at recombination, $T(t_0)/T(t_{\text{rec}})$, is about 10^{-3} , which means that the background radiation has been redshifted by a factor of about 1000 since the time of recombination.

Anisotropy of the cosmic background radiation and implications for models of large-scale structure.

Information about the isotropy of the cosmic background radiation is crucial to studies of structure formation because it allows us to infer the size of density fluctuations at the recombination time, when the cosmic background radiation decoupled from baryonic matter. If the mass distribution at the time of decoupling had been totally uniform, the cosmic background radiation would be isotropic, or the same in all directions. However, in 1992 another instrument on COBE detected inhomogeneities; the temperature of the background radiation varies depending on the direction the radiation comes from. The temperature differences are quite small: $\delta T/T \approx 6 \times 10^{-6}$. Those differences are believed to reflect density inhomogeneities present at the time of recombination; photons emanating from a high-density region must have lost energy in “climbing out” of the region and thus suffered a redshift over and above the cosmic redshift. Fluctuations in the cosmic background radiation

were seen at all angular separations larger than about 10 degrees of arc (the resolution of the relevant instrument on COBE).

Ten degrees of arc corresponds to hundreds of megaparsecs now and to hundreds of kiloparsecs at the time of recombination, a distance larger than the Hubble radius, $R_H \equiv c/H(t)$, at that time. Thus the fluctuations measured by COBE had wavelengths larger than the Hubble radius (or the horizon) when they were imprinted on the cosmic background radiation. They therefore were apparently not connected causally at the time of the imprinting. In that case the near anisotropy of the background radiation is very difficult to understand. This paradox is an aspect of the “horizon problem” pointed out by Misner in the 1960s. The problem is solved by postulating a period of inflation, or exponential expansion of the very early universe. In inflationary scenarios, regions that now appear causally disconnected were in fact causally connected prior to inflation. The point of this discussion is that the fact that the scale of the observed temperature fluctuations is larger than the Hubble radius at t_{recomb} implies that these large-scale fluctuations could not have been affected by small-scale physics after inflation and must indeed carry information about primordial fluctuations.

Since the shape of the CDM spectrum of fluctuations at the time of recombination can be calculated from the assumptions of the CDM model, the observed amplitudes determine the normalization (amplitude) of the spectrum on all scales. Had the measured fluctuations been much smaller in amplitude than 6×10^{-6} , not only the CDM scenario, but also the idea that gravity is the decisive force responsible for structure formation, would have been in jeopardy. Not only are the COBE temperature-fluctuation measurements thus

consistent with the CDM model, but as described in the section “Cold Dark Matter, Large Scales, and COBE” in the main text, they also fix the value of the only undetermined parameter of the model. \square

Further Reading

E. Bertschinger. 1994. Cosmic structure formation. *Physica D*, in press.

H. P. Gush, M. Halpern, and E. H. Wishnow. 1990. Rocket measurement of the cosmic-background-radiation mm-wave spectrum. *Physical Review Letters* 65: 537–540.

J. C. Mather et al. 1994. Measurement of the cosmic microwave background spectrum by the COBE FIRAS instrument. *The Astrophysical Journal* 420: 439–444.

P. J. E. Peebles. 1993. *Principles of Physical Cosmology*. Princeton University Press.

Michael S. Turner. 1987. A cosmologist’s tour through the new particle zoo (candy shop?). In *Dark Matter in the Universe*, edited by J. Kormendy and G. R. Knapp. Reidel.

Terry P. Walker, Gary Steigman, David N. Schramm, Keith A. Olive, and Ho-Shik Kang. 1991. Primordial nucleosynthesis redux. *The Astrophysical Journal* 376: 51–69.

Steven Weinberg. 1972. *Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity*. John Wiley and Sons.

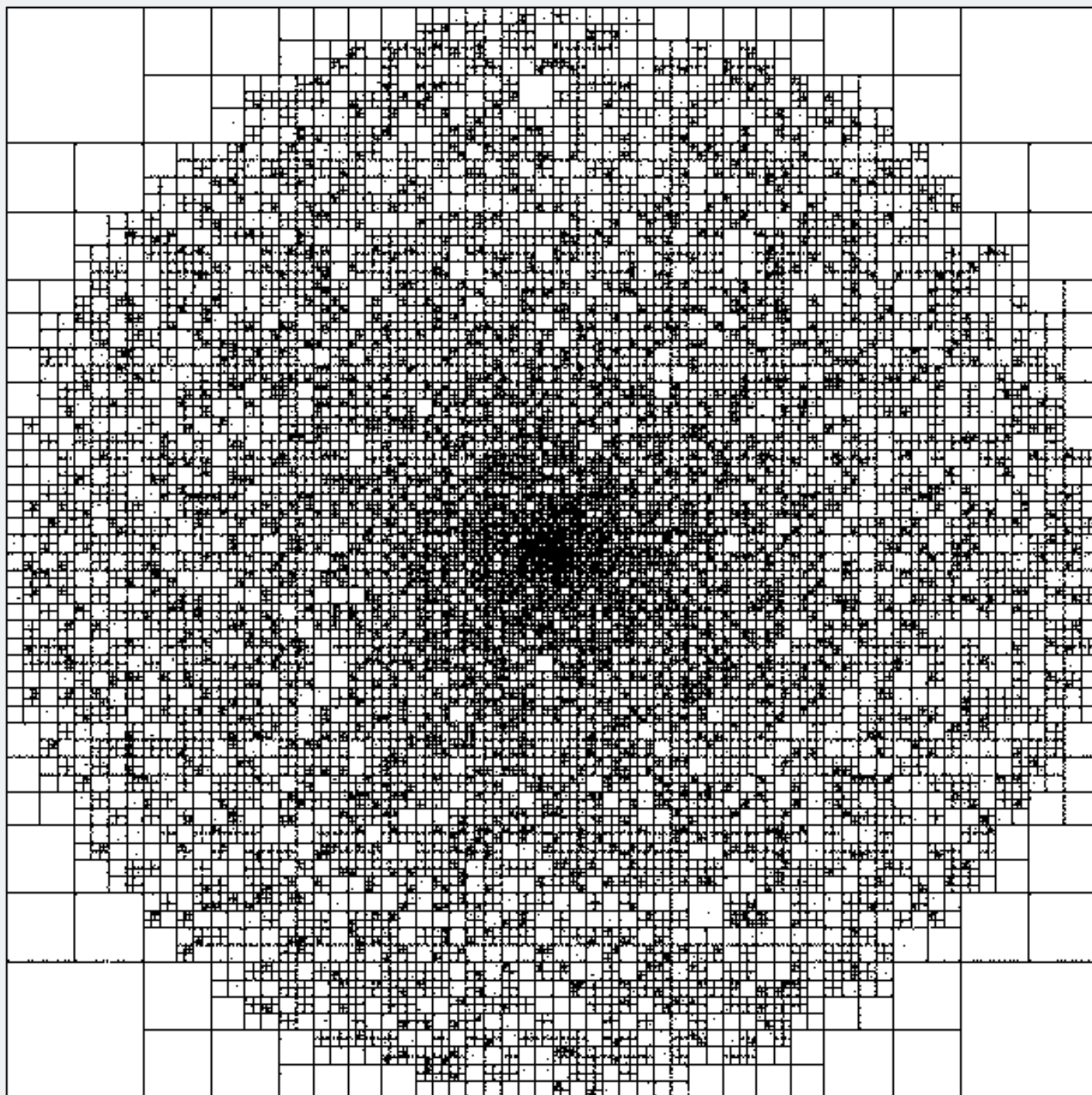
Steven Weinberg. 1977. *The First Three Minutes: A Modern View of the Origin of the Universe*. Basic Books.

Salman Habib performed his undergraduate work at the Indian Institute of Technology, Delhi, and his graduate work at the University of Maryland, College Park. He now has a joint postdoctoral fellowship in the Theoretical Astrophysics and the Elementary Particles and Field Theory groups at the Laboratory. His research interests are quantum field theory, statistical mechanics, nonlinear dynamics, cosmology, and astrophysics.

Raymond J. Laflamme received his bachelor’s degree from the Université Laval in his native Canada and his Ph.D. from Cambridge University under the direction of Stephen Hawking. He is now an Oppenheimer fellow in the Theoretical Astrophysics Group, studying cosmology, the early universe, and quantum mechanics.

A Fast Tree Code for Many-Body Problems

Michael S. Warren and John K. Salmon



Parallel computing is bringing about a revolution in the study of the large-scale structure in the universe. Popular models, such as the cold-dark-matter model discussed in the main article, assume that a nearly homogeneous initial matter distribution has evolved through gravitational interactions to the froth-like distribution of galaxies observed in the night sky. But which of the proposed models actually predicts a matter distribution that matches those observations? Cosmologists use analytical methods to predict how the matter distribution evolves up to the time when nonlinear growth becomes important. If those preliminary results do not rule out a particular model, cosmologists turn to computer simulations to follow the nonlinear growth of structure. Only now, however, through the use of the latest massively parallel computers, can the simulations include enough particles to resolve simultaneously the many scales relevant to the problem.

We have developed a fast tree code that can handle tens of millions of particles and used it in the first high-resolution test of the cold-dark-matter model. The simulations were run on the Intel Touchstone Delta, a prototype massively parallel machine; results are presented in "Experimental Cosmology and the Puzzle of Large-Scale Structure." Here we discuss the innovative aspects of our tree code and its applicability not only to cosmology but also to a wide range of hydrodynamic and other many-body problems.

The challenge of simulating millions of particles as they move under the influence of mutual gravitational attraction is quite formidable. Because gravity is a long-range force (falling off only as the square of the distance), an exact algorithm would require calculating the force between each pair of particles at each timestep of the simula-

tion. If there are N particles, there are $(N^2 - N)/2$ pairs. Thus when N is large, the computation time for the force calculation is proportional to N^2 —in the language of computer science, the time is $O(N^2)$. A brute-force simulation involving millions of particles would have required months on what is currently the fastest computer at Los Alamos (and arguably the fastest computer in the world), the 1024-processor CM-5 Connection Machine. Fortunately, there are approximation methods known as hierarchical tree methods that reduce the time needed for the force calculation from $O(N^2)$ to $O(M \log N)$ or $O(N)$, a dramatic reduction when N is in the millions.

Hierarchical tree methods were invented in 1985 and have been applied successfully to simulations of large-scale structure on scalar and vector computers. However, all those simulations lacked resolution on the desired range of scales desired for cosmology. Our particular challenge was to implement a tree method to simulate millions of particles on a massively parallel computer.

Three main difficulties arise in adapting tree codes to such a computer. First, the problem must be divided into similar or identical parts, one for each processor in the parallel machine, such that the parts require roughly equal amounts of computation; in computer jargon, the processors should be load-balanced. A standard method for problems of this type is to divide space into regions called processor domains and have each processor perform the calculations for the particles in the domain assigned to it. However, since the particles in cosmological simulations are distributed irregularly, there is no a priori division of space into domains that are load-balanced by virtue of having roughly equal numbers of particles. Moreover, since the particles move

with respect to one another, domains that are initially load-balanced do not necessarily remain load-balanced.

The second difficulty relates to communication between processors. Most currently popular parallel computers have a distributed memory; that is, each processor is connected to its own memory, which stores the data for its domain. Communication between one processor and another is slow. Unfortunately, because we are studying long-range forces, considerable communication between processors is inevitable. If the communication is not minimized, the program will take prohibitively long to run.

The third difficulty relates to how the instructions for interprocessor communication are written. "Data-parallel" languages such as FORTRAN 90 hide the details of interprocessor communication from the programmer only when the same steps are performed simultaneously on a fixed large number of data elements, as in arithmetic involving large arrays. Since the calculations for particles in tree codes depend on how many other particles are nearby, the communication instructions must be programmed explicitly. Of the few calculations achieving high performance on massively parallel computers, most have been relatively regular and static problems that did not present difficulties comparable to those presented by complex many-body problems.

We have overcome these difficulties by applying what is called a key scheme for storing and retrieving data. That scheme provides an efficient way to describe both the particle locations and the organization of particle data into a computational tree. The key scheme, along with other innovations described below, produces a considerable advance in computing speed and resolution in the solution of N -body problems. For example, we know of

another astrophysical simulation, which was run on an IBM vector supercomputer, that used approximately the same number of particles as ours (marginally fewer). Our simulation provided twenty times greater spatial resolution than that simulation while requiring only a twentieth of the computation time.

This article will discuss our tree code, focusing on a few aspects that can be described briefly. A more complete discussion appears in our paper “A Parallel Hashed Oct-Tree N-Body Algorithm,” listed in the Further Reading.

We have taken care to produce a “friendly” code. Each part of the calculation is performed by a different section of the code, and the sections are as independent of one another as possible. The modularity was achieved through a large expenditure of programming time—we had to start from scratch, rather than modify a less well-organized program that we had originally written for sequential computers. But the modular structure has a large payoff: The code is easily adapted to solve other N -body problems. As the few modules specifically determined by cosmology can easily be replaced with modules describing interactions other than gravity, users do not need to be familiar with the details of the code that deal with parallel computation. An additional payoff is portability. We have developed modules for the machine-dependent parts, such as input-output and interprocessor communication of the program that allow it to run on computers ranging from ordinary sequential machines to clusters of workstations to massively parallel machines such as the CM-5 Connection Machine.

Our program is now serving several purposes. It has been indispensable in performing statistical analyses and data processing on the output of our simulations, since their size prohibits analysis on anything but a parallel supercomput-

er. We have written a module that can calculate both the three-dimensional dynamics of compressible fluids using smoothed-particle hydrodynamics (with or without gravity) and have adapted it to do three-dimensional incompressible hydrodynamics by a vortex-particle method as well. We plan to use smoothed-particle hydrodynamics to investigate galaxy formation, a critical step in connecting our cosmological studies to observations.

Our code can be applied to a wide variety of problems where long-range pairwise interactions dominate the computational cost. In addition to the areas mentioned above, accelerator beam dynamics, computational biology (protein folding), chemistry (molecular structure and thermodynamics), electromagnetic scattering, fluid mechanics with the panel method (commonly used to design subsonic airplanes), molecular dynamics, and plasma physics are those we know of, and there are certainly more. We are establishing collaborations with other researchers who we hope can successfully apply our code to current problems in a number of those fields. In addition, we are testing the efficiency of different computational approaches. Our N -body program may be unique in that it is being used both as a production astrophysics code and as a testbed for algorithms and interdisciplinary applications.

Computational Methods

The overall structure of the program is straightforward. First the net force on each particle from the others is calculated. Then the position and velocity of each particle at a slightly later time are computed from that force and its present position and velocity. This procedure (called a timestep) is repeated as often as the user wants or has computer

time for. As mentioned above, the force calculation is the time-consuming part. Here we discuss our methods for reducing the amount of time consumed by the force calculation.

The tree method. The tree method is a way to take advantage of the basic approximation method of our program: the multipole expansion. A group of particles at a distance exerts almost the same force as a large single particle at the group’s center of mass; approximating the group as a single particle is equivalent to using only the monopole term in the multipole expansion. However, when the group of particles is close to the particle on which the force is being calculated (especially when the distance is small compared to the size of the group), the monopole approximation is less accurate. In that situation, one can improve the approximation by using higher terms in the multipole expansion. Our program offers that option, but we have found that another approach leads to a faster program. We improve the approximation by using the basic idea of tree codes: dividing the group into smaller groups. Then each of the smaller groups can be treated as a single particle.

In calculating the force from a group on a given particle, one should divide the group finely enough to obtain good accuracy. However, dividing it more finely than necessary results in more calculations than necessary. To divide each group in an efficient way for each particle, one can set up a “tree,” a hierarchy of finer and finer levels of detail, and use the coarsest acceptable level. In tree codes, space is divided hierarchically into a tree of cells. Figure 1 shows a two-dimensional tree analogous to the three-dimensional trees in our astrophysical simulation. The largest cell, the root of the tree, is the entire region of space. That cell is di-

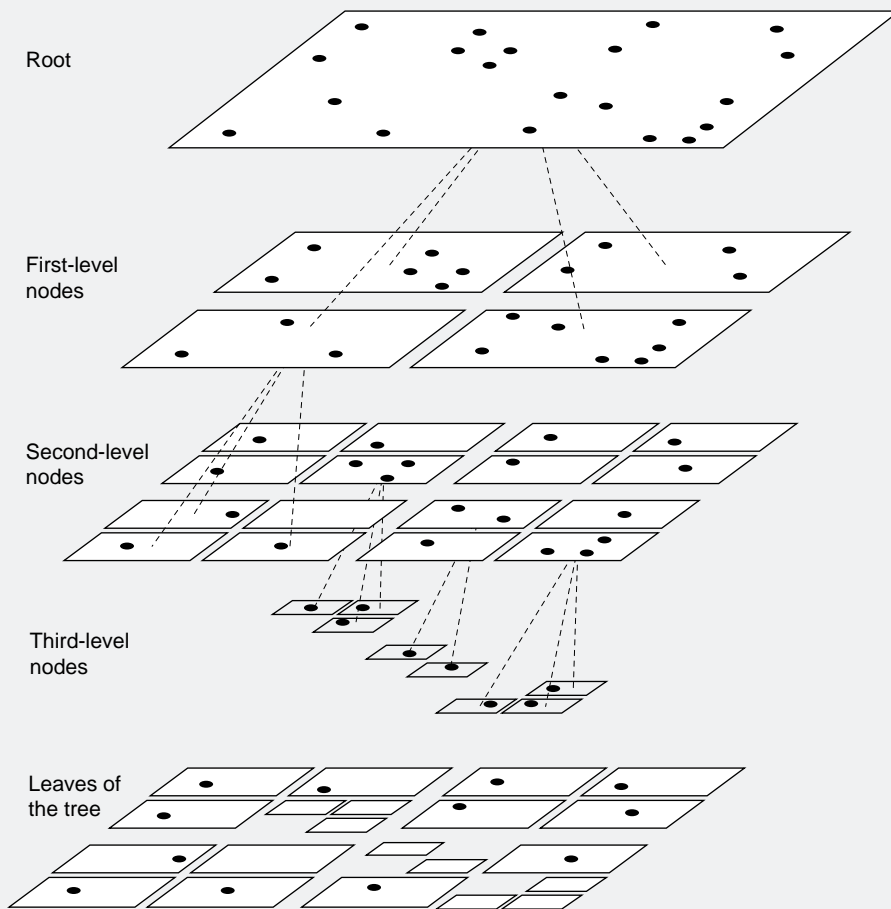


Figure 1. A Regular Two-Dimensional Tree

The tree has been produced by dividing space into square cells so that each particle is in a separate cell. The branches of the tree connect each square to the smaller squares or rectangles made by cutting through its center. The smaller squares in turn can branch out by being further divided. The leaves are the squares and rectangles that contain exactly one particle and therefore do not need to be divided further. At the bottom is a “flat” representation of the tree structure induced by the particles. Since the cells are squares and are divided orthogonally through the center, a cell can have at most four daughters. Such a tree is called a quad-tree. The analogous three-dimensional trees in our cosmological simulations are called oct-trees.

vided into smaller “daughter” cells, and they are in turn divided into smaller cells, and so forth. Cells containing one particle are not divided; they are the leaves of the tree. (Areas of space containing no particles are ignored.) Thus the structure of the tree adapts to the positions of the particles, having

many levels of refinement where particles are densely clustered. The structure must be recalculated every time the particle positions are updated.

Our tree method must include a criterion for determining when to approximate the force due to a group of particles as a force due to a monopole at the

group’s center of mass. The minimum distance is called the critical radius, r_c . The method of calculating the critical radius, or “multipole-acceptance criterion,” is crucial to the speed and accuracy of the tree code. As described below, we calculate the r_c of a group of particles from its distribution—in particular, from its higher multipole moments, which are precisely the terms discarded by the monopole approximation—in such a way that the error of the monopole approximation is less than a limit set by the user.

To calculate the force on a given particle from the others, one “traverses” the tree node by node, starting at the root and going to finer and finer levels of detail. As illustrated in Figure 2, whenever the monopole approximation is acceptable, one skips all daughters and further descendants of that cell, thus saving the time that would be needed to calculate the force from each particle in that cell individually. If instead the cell is closer than r_c , one repeats the process with each of the cell’s daughters, each of which has its own r_c . The process of examining smaller and smaller cells can continue until forces from individual particles are calculated, if necessary. The execution time of the tree traversal described here is $O(N \log N)$. The enormous speedup when N is large justifies spending a relatively small amount of time in the program to rebuild the tree at every timestep—and also justifies our spending a good deal of our own time in developing the program.

Keys. A structure as complex as a tree is difficult to implement on a parallel computer. The description of the tree, which includes the coordinates of each cell and properties such as center-of-mass location and multipole moments, must also provide a way of finding the daughters of any cell so that the

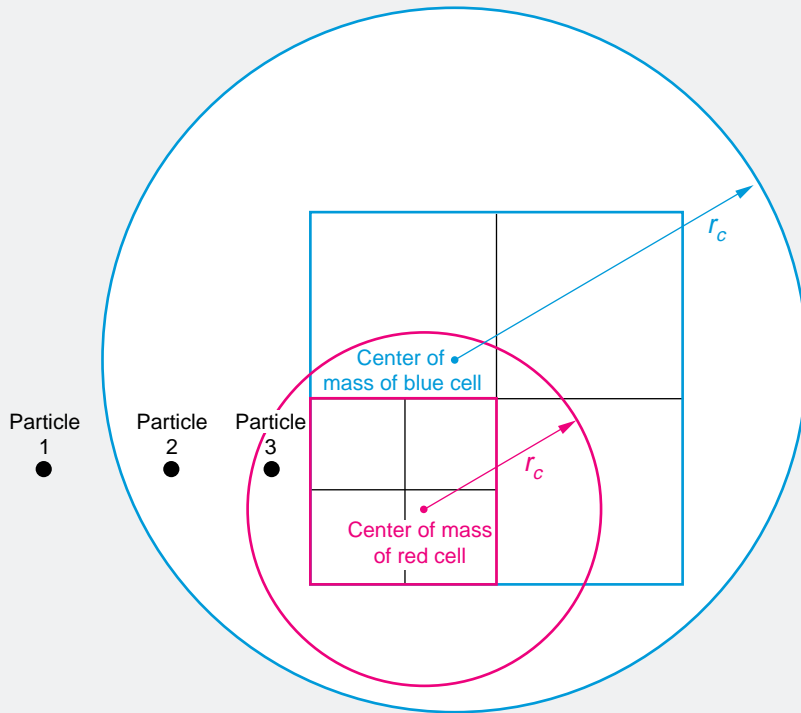


Figure 2. The Multipole-Acceptance Criterion

When the distance between a particle and a cell is greater than the critical radius of the cell, all the mass in the cell can be treated as a single point mass; that is, the force exerted by the particles in the cell can be calculated with the use of the monopole approximation. The radii of the blue and red circles are the critical radii of the cells shown in blue and red respectively. Particle 1 is outside the critical radius of the cell shown in blue, so the monopole approximation is used to find the force on particle 1 due to all the particles in the blue cell together. Particle 2 is inside the critical radius of the blue cell, so the monopole approximation is not applied to the force exerted on it by the particles in the blue cell. However, since the particle is outside the critical radius of the red cell, the monopole approximation is used to find the force exerted on it by the particles inside the red cell (and similarly for the other three daughters of the blue cell). Finally, particle 3 is inside the critical radius of the red cell, so the monopole approximation can be used only for the force exerted on it by even smaller cells (with boundaries in black) within the red cell.

tree can be traversed. Trees are most often described by storing the addresses of each cell's daughter cells along with the data (such as the mass) for the cell. Those addresses are called pointers to the daughter cells.

The pointer method has two disadvantages for parallel programs. First, since each pointer is dynamically deter-

mined by such considerations as the address of the next available memory location, the memory location the pointer points to has nothing to do with the spatial location of the cell. Second, when one processor must retrieve information about cells in the domain of another processor, the pointers in a parent cell in one processor must be somehow

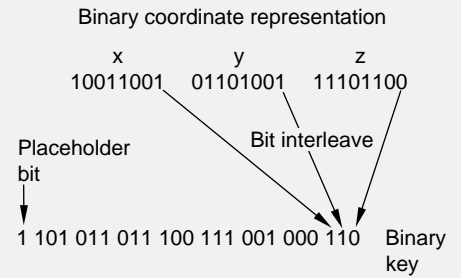


Figure 3. The Key Mapping

The key for each particle is generated from its coordinates measured from a corner of the region of space (a cube). The bits of the coordinates are interleaved and a 1-bit is attached to the beginning of the key as a placeholder, to distinguish particle keys from cell keys. The key derived from three single-precision floating-point numbers fits nicely into a single 64-bit integer or a pair of 32-bit integers. In this example, the 8-bit x, y, and z coordinates are mapped to a 25-bit key.

translated into valid addresses of daughter cells in another processor.

One solution is for each processor to retrieve all the data it could possibly need at an early stage in the program. It can then build its own "private" copy of the tree structure, and proceed with the rest of the calculation without having to worry about where the data are (because it has already guaranteed it has all the data it will need). In fact, our first version of a parallel tree code worked in that way. However, determining beforehand which data are needed can be somewhat complicated. In our current program the multipole-acceptance criterion requires knowing the contents of each particular cell (which aren't known until the tree is traversed). In this case, an easier communication method is for each processor to ask for data when they are needed, not before. This method requires a mechanism to "ask" for each piece of data and retrieve it from another processor in an efficient manner.

The identifier of each particle in our simulation is a key derived from the particle's coordinates, as shown in Figure 3. We translate keys into addresses

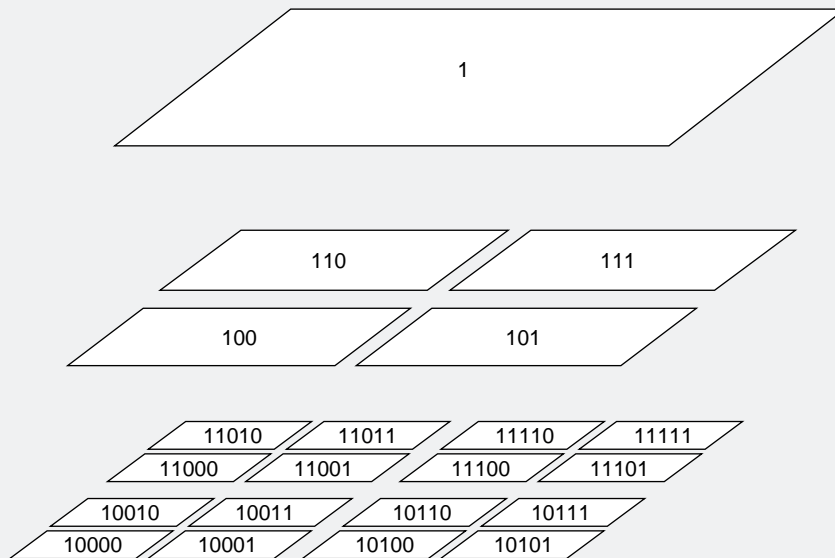


Figure 4. Cell Keys

The figure shows a two-dimensional tree together with the key associated with each node. The keys are generated by interleaving bits of the coordinates in the same way that particle keys are generated from particle coordinates. In fact, the coordinates of each cell are the initial bits that the coordinates of all the particles in that cell share. When particles are very close together, cell coordinates may have as many bits as particle coordinates. In order to distinguish the higher-level nodes of the tree from the lower-level nodes, we attach an additional 1-bit to the most significant bit of every key (the place-holder bit). Without the place-holder bit, there would be no distinction between the keys 11 and 000011, for instance. The root node is represented by the key 1.

of cell data through a standard technique called hashing. Given the key of a particle, the data for the particle can be rapidly retrieved, even by one processor from another; the key scheme provides a uniform addressing mechanism. We use a similar scheme to generate the address of each cell from its coordinates, as shown in Figure 4. Here the key scheme allows us to code tree traversals as simply as when using pointers, because we can find the keys of daughter or parent cells by performing simple bit arithmetic on the key of a cell. It also allows us to find any node of the tree in time that is $O(1)$, that is, independent of N . (In contrast, if we want to find a particular node of a tree whose topology is described by

pointers, we must start at the root of the tree and traverse until we find the desired node, which takes $O(\log N)$ time.)

The key method is particularly advantageous when we sort the particles in the numerical order of their keys. Because of the way the keys depend on the coordinates, particles whose addresses are near each other in the sorted list are usually near each other in space. That property is useful in several parts of the program, as we shall see below.

Features of the Code

The organization of our program is sketched in Figure 5. Each part must

be made to work efficiently on a massively parallel computer. Techniques for doing so are discussed in “A Parallel Hashed Oct-Tree N-Body Algorithm.” Here we discuss a few techniques that can be described simply.

The multipole-acceptance criterion and analytic error bounds. The multipole-acceptance criterion is crucial to the accuracy and efficiency of the program. Our criterion rejects all particle-cell interactions where the approximation would introduce large errors, while accepting as many interactions as possible where the error is small so as to avoid unnecessary computation. Until now, multipole-acceptance criteria in many-body simulations have incorporated information about the size of a cell, but no information about its contents other than the position of its center of mass. Some of them also had the disadvantage that the worst-case errors arising from the approximation were unbounded, and the errors in realistic situations could be quite large. The popular “fast multipole method” does have a well-defined worst-case error bound, but has proved to be quite slow in three dimensions.

We developed a multipole-acceptance criterion that directly depends on the contents of the cell, specifically, on the largest distance of a particle in the cell from the cell’s center of mass and on the first two multipole moments discarded by the approximation—the dipole and quadrupole moments, when we keep only the monopole term in the expansion. (The calculation of r_c is then particularly simple; because negative masses don’t exist, the dipole moment of any mass distribution about its center of mass vanishes.) Because the criterion depends on the information discarded by the approximation, we can set an error bound and know that our use of the multipole approximation

introduces no errors larger than that bound. We have tested the speed and accuracy of our multipole-acceptance criterion using several sets of initial conditions. Our criterion moderately decreased the root-mean-square error and decreased the maximum error by factors ranging from 3 to 10. Details are given in the article "Skeletons from the Treecode Closet," listed in Further Reading.

Parallel data decomposition. The parallel data decomposition is critical to the performance of a parallel algorithm. A conceptually simple and easily programmed method may result in unacceptable load imbalance. A method that attempts to balance the work precisely may take so long that performance of the overall program suffers.

Our method is to cut the list of particle keys into a number of pieces equal to the number of processors. The divisions are placed so that the pieces require equal total amounts of work; the work for each particle is readily approximated by counting the number of cells and particles the given particle interacted with on the previous timestep. The method tends to produce processor domains that consist of spatially grouped particles. The grouping greatly improves the efficiency of the traversal stage of the algorithm, since the amount of data needed from other processors is roughly proportional to the surface area of the processor domain. Figure 6 illustrates how this divides a centrally clustered two-dimensional set of particles among 16 processors. One source of inefficiency is that our method of generating keys from coordinates creates a number of spatial discontinuities in the sorted list. A processor domain can span one of those discontinuities and thus consist of two spatially separated groups of particles. We have used a different order-

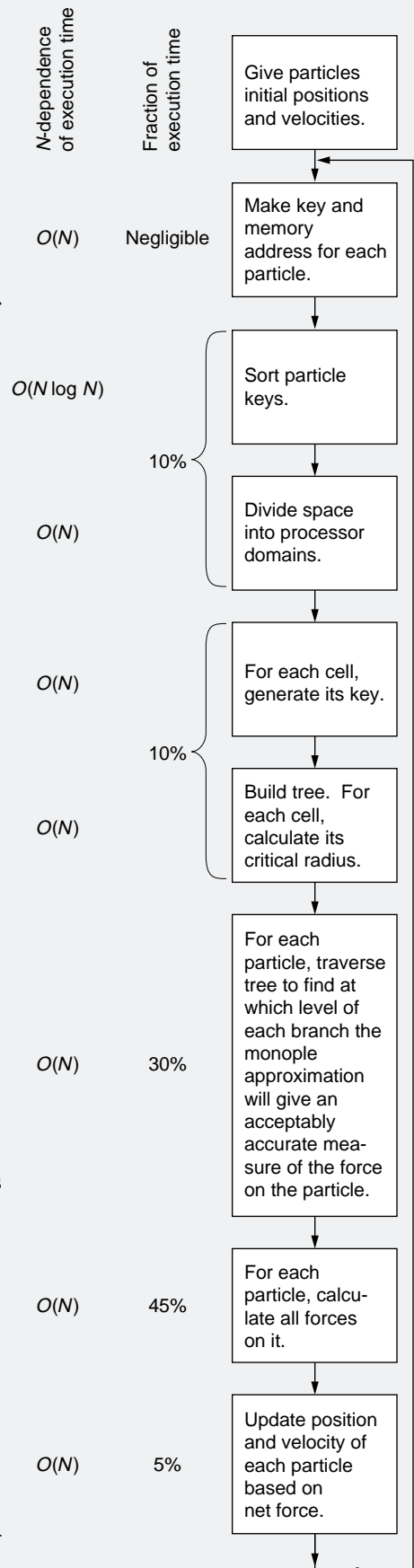
Figure 5. The Organization of our Algorithm

All parts are done in parallel. After the initialization of particle positions and velocities, the process is repeated the number of times specified by the user—usually hundreds or thousands of timesteps.

ing, which does not contain discontinuities, but it improves performance only slightly.

Tree construction. Sorting the particle keys is also advantageous in tree construction. In fact, our original reason for sorting these identifiers was to improve the tree-construction stage. In the usual algorithm for constructing a tree, each particle is inserted at the root of the partially constructed tree. The algorithm determines which of the top-level cells includes its position, then which of that cell's daughters, and thus the particle moves downward one node at a time until a new cell is created to be its leaf. That process is $O(\log N)$ for each particle. In our code, however, the particles are added to the tree in the sorted order, and each one is inserted not at the root but at the location of the last particle inserted. Since particles near each other in the sorted list are usually near each other in space, moving a particle to its correct location in the tree is now on average $O(1)$. This single increase in efficiency makes up for the time spent in sorting.

Memory hierarchy and access patterns. Tree codes place heavy demands on the memory subsystems of modern computers because the amount of data that must be transferred between processors is large and not well-ordered. We have encouraged a more orderly and efficient memory-access pattern by arranging the order of computation to take advantage of the underlying structure of the algorithm. In



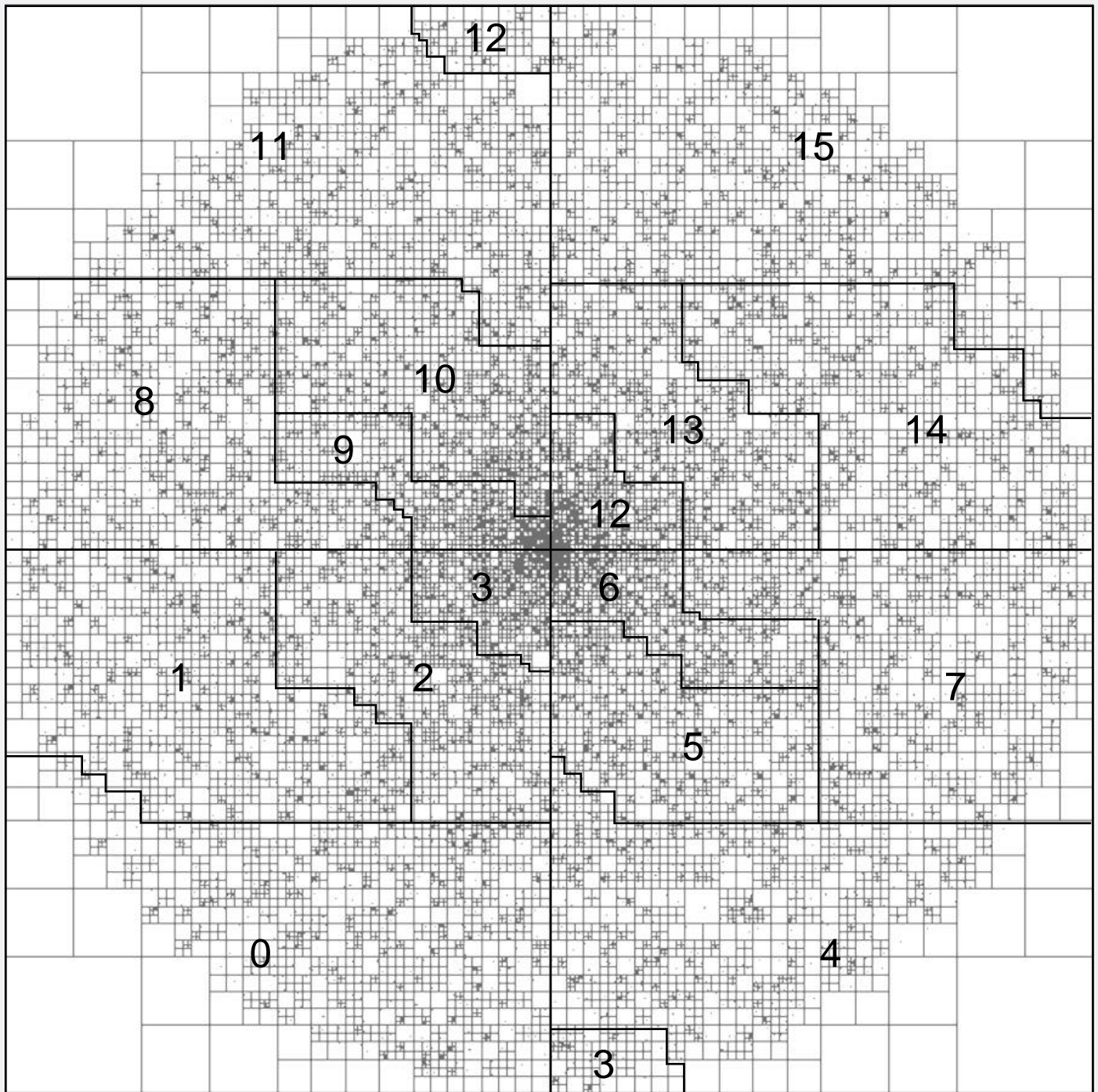


Figure 6. Data Decomposition

Shown are the processor domains assigned to all 16 processors by a data decomposition for a clustered system of particles. Each domain shown is the smallest that contains all the particles in a section of the sorted particle-key list. The compactness of the domains reduces the amount of interprocessor communication required in evaluating the interaction of the particles in the domain.

particular, the computation runs fastest when each processor retrieves most of the data it uses from the memories to which it is connected directly. We wish to keep data for as long as possible in the fastest level of the hierarchy

that comprises registers, cache, local memory, other processors' memory, and virtual memory. A helpful property of tree algorithms is that particles that are near each other tend to interact with almost the same sets of cells; thus

the calculations of those interactions require almost the same data. By updating the particles' positions in the order of the sorted key list, we greatly reduce the fraction of slow memory accesses. On parallel computers we could extend

this “virtual cache” model even further by erasing from each processor’s memory information from other processors that has not been used recently. We expect that implementing this technique will allow significantly larger simulations to take place by eliminating copies of cells from other processors (which currently uses the largest amount of memory, and thus limits the maximum size of the simulation).

Portability. A portable program is one that can be run on a variety of computer systems after only a small amount of time is spent changing the program for each particular computer. In contrast, a non-portable program may take nearly as long to rewrite for each system as it took to write in the first place. Computational scientists would rather spend that time solving new problems or writing better algorithms.

In the modern world of rapidly evolving parallel architectures, a particular brand of computer may be the best available for a time measured in months. If one wants to solve physics problems on the best parallel computers, one does not have much time to get a program working on each new model. Thus portability increases the problem-solving efficiency of the combined system of scientist, programmer, and computer. However, one must always keep in mind the tradeoffs between portability and other desirable features of a program (such as speed). In some cases the problem-solving efficiency is increased instead by methods (such as writing often-executed inner loops in a particular computer’s assembly language) that sacrifice some portability for increased speed.

A concept related to portability is the “adaptability” of a program. One would rather not solve almost the same programming problem time after time.

The addition of more complex physical laws to a simulation and the application of it to very different physical problems that have a similar computational structure are common needs. If one can isolate individual parts of a program as modules that they can be easily replaced without affecting the behavior of the other modules in the program, one can write software that is much more adaptable to different problems.

We have tried very hard to design our code to meet the dual goals of portability and adaptability, which present an even greater challenge than usual when one is using parallel machines. Our efforts have resulted in a code that allows scientists in many disciplines to benefit from the enormous speedup offered by tree methods, not only to solve problems on supercomputers that were simply insoluble before (such as our cosmological problems), but also to perform on workstations computations that formerly required supercomputers. We have adapted the code to run on many different computers, from workstations to the latest massively parallel machines. That adaptation has required the definition of a few “standard” message-passing functions that can be easily implemented on any type of distributed memory parallel computer. To adapt the code to a new machine, one need only change the implementations of a few functions in a single file. We have also implemented these calls with the standard Parallel Virtual Machine (PVM) library and the newly established Message Passing Interface (MPI) standard. Thus, the program should run without modification on any machine that implements these standards. We hope that our code can become a successful model for a “standard library” that can be used by scientists who do not have (or do not want) a detailed knowledge of how parallel computers work.

Performance

We timed the various stages of the algorithm on the 512-processor Intel Touchstone Delta installed at Caltech and partially owned by the Laboratory, which is a prototype of Intel’s Paragon supercomputer. The timings are from an 8.8-million-particle production run simulating the formation of structure in a cold-dark-matter universe. During the initial stages of the calculation, the particles are spread uniformly throughout the spherical computational volume. We set an absolute bound on the error of the acceleration due to each interaction; the error bound is 10^{-3} times the mean acceleration of particles in the simulation. This bound results in 2.2×10^{10} interactions per timestep in the initial unclustered system. At this stage the program runs at 5.8 billion floating-point instructions per second (gigaflops).

Computation Stage	Time (s)
Domain Decomposition	7
Tree Building	10
Tree Traversal	33
Data Communication	6
Force Evaluation	54
Load Imbalance	7
Total (5.8 Gflops)	114

In later stages of the calculation the system becomes extremely clustered—the density in large clusters of particles is typically 10^6 times the mean density. The number of interactions required to maintain the same accuracy grows moderately as the system evolves. At a slightly increased error bound of 4×10^{-3} , the number of interactions in the clustered system is 2.6×10^{10} per timestep. At this stage the program runs at 4.9 gigaflops.

Computation Stage	Time (s)
Domain Decomposition	19
Tree Building	0
Tree Traversal	55
Data Communication	4
Force Evaluation	60
Load Imbalance	12
Total (4.9 Gflops)	160

Almost half of the execution time is spent in the force-calculation routine. This routine consists of a few tens of lines of code, so it makes sense to obtain the maximum possible performance through careful tuning. For the Delta's i860 microprocessor we hand-coded the force-calculation routine in assembly language. The resulting routine runs at a speed of 28 megaflops per processing node.

If we count as "useful work" only the floating-point operations performed in the force-calculation routine (30 flops per interaction) the overall speed of the code is about 5–6 gigaflops. However, this number is in a sense unfair to the overall algorithm, since the majority of the code is not involved in floating-point operations, but in tree traversal and data-structure manipulation. The integer-arithmetic and addressing speeds of the processor are as important as the floating-point performance. We hope that evaluation of processors does not become overbalanced toward floating-point speed at the expense of integer arithmetic and memory bandwidth. Our code provides a good example of why a balanced processor architecture is necessary for good overall performance.

Conclusion

The code described here is by no means a "final" version. The imple-

mentation has been explicitly designed to easily allow experimentation and inclusion of new ideas that we find useful. We will continue to use it not only to study the process of galaxy formation, but also to investigate multipole algorithms. We have been studying the addition of cell-cell interactions (similar to those used in the fast multipole method), which reduces the N -dependence of the algorithm from $O(N \log N)$ to $O(N)$. Cell-cell interactions are performed by approximation of the gravitational field at each particle in a cell by Taylor expansion about the center of the cell. Preliminary results indicate that this new method reduces the number of interactions by a factor of 5 in a simulation of 1 million bodies.

In an overall view of this algorithm, we feel that two general points deserve special attention:

- The fundamental ideas in this algorithm are, for the most part, standard tools of computer science (key mapping, hashing, sorting). In combination, they form the basis of a clean and efficient parallel algorithm. Such an algorithm does not evolve from a sequential method. It requires starting anew, without the prejudices inherent in a program (or programmer) accustomed to using a single processor.
- The computing speed of the code on an extremely irregular, dynamically changing set of particles that require global data for their update, using a large number of processors (512), is comparable with the performance quoted for much more regular and static problems, which are sometimes identified as the only type of "scalable" algorithms that obtain good performance on parallel machines. We hope we have convinced the reader that even difficult irregular problems are amenable to parallel computation.

We expect that algorithms like those described here, coupled with the extraordinary increase in computational power expected in the coming years, will play a major part in the process of understanding complex physical systems. □

Further Reading

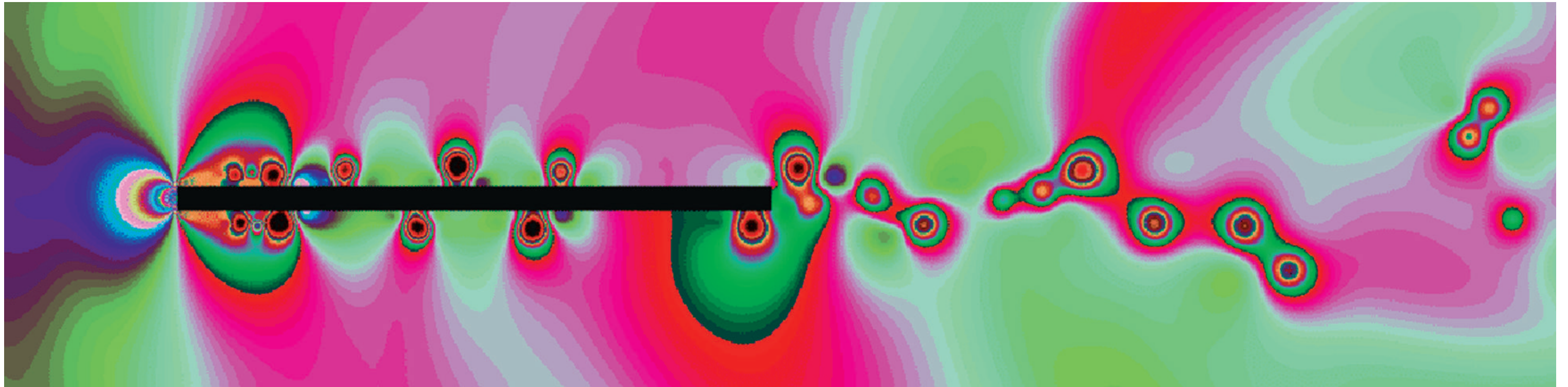
John K. Salmon and Michael S. Warren. 1992. Skeletons from the treecode closet. *Journal of Computational Physics* 111:136–155.

John K. Salmon, Michael S. Warren, and Grégoire S. Winckelmans. 1994. Fast parallel treecodes for gravitational and fluid dynamical N-body problems. *International Journal of Supercomputer Applications* 8, in press.

Michael S. Warren and John K. Salmon. 1993. A parallel hashed oct-tree N-body algorithm. In *Supercomputing '93*. IEEE Computer Society Press.

John K. Salmon is a research fellow in physics at California Institute of Technology. He holds a B.S. in physics and a B.S. in electrical engineering and computer science from Massachusetts Institute of Technology, an M.S. in physics from the University of California, Berkeley, and a Ph.D. in physics from the California Institute of Technology. His research interests include applications of parallel fast particle methods to problems in astrophysics, computational fluid dynamics and other areas of computational science. With Michael Warren, he won the 1992 Gordon Bell Prize for achievement in high-performance computing.

The biography of co-author Michael S. Warren appears on page 81.



Lattice-Boltzmann
a versatile tool for multiphase

Fluid Dynamics
and other complicated flows

Shiyi Chen, Gary D. Doolen, and Kenneth G. Eggert



Long-term research efforts at the Laboratory often produce results that prove highly valuable to U.S. industry. The example described here is a new approach to modeling the flow of multiphase fluid mixtures, such as oil and water, through very complicated geometries. This new modeling technique, called the lattice-Boltzmann method, evolved out of ideas that have been intensely investigated since 1985, when Laboratory scientists and others discovered that very simple models of discrete particles confined to a lattice can be used to solve very complicated flow problems.

This lattice method can be regarded as one of the simplest microscopic, or particle, approaches to modeling macroscopic dynamics. It is based on the Boltzmann transport equation for the time rate of change of the particle distribution function in a particular state. The Boltzmann equation simply says that the rate of change is the number of particles scattered into that state minus the number scattered out of that state.

The method is fully parallel (the same calculations are performed at every lattice site) and local (only nearby particles interact with each other). It is therefore easily programmed and runs efficiently on parallel machines. Complex boundary conditions are incorporated in a straightforward way and cause the calculational speed to decrease by only a few percent. The

Illustration on previous spread: A lattice-Boltzmann simulation of flow past a slab shows the complicated flow patterns that can be modeled with this technique. The lattice is 1024×256 sites; the thickness of the slab is 32 sites, which gives the flow a Reynolds number of 960. Wind-tunnel boundary conditions are used—the flow velocity at the entrance (left), the top, and the bottom boundaries is fixed. The top image shows contours of equal vorticity; the bottom image shows the pressure distribution of the flow. The simulation was done using the Connection Machine 2 at the Advanced Computing Laboratory.

method yields a good approximation to the standard equations of fluid flow, the Navier-Stokes equations, in the limit of long wavelengths and low frequencies. Also, recent generalizations of the method have extended its applicability to multiphase flows, chemically reacting flows, diffusion and thermohydrodynamics.

United States oil companies have expressed considerable interest in the lattice-Boltzmann method for addressing problems in oil recovery. For example, we are collaborating with the Mobil Exploration and Producing Technical Center on using the lattice-Boltzmann method to simulate the flow of oil and water through oil-bearing sandstone at a scale and an accuracy never before possible. Mobil provided Los Alamos with 5-micron-resolution sandstone geometries for use in the simulations. (Typical pore diameters are tens of microns.) The goal of the work is to simulate, at the scale of individual pores in the rock, what happens when water is pumped through the rock to force out oil. Viscosities, surface tensions, contact angles, and surfactant effects can be varied to determine how well this method of oil recovery can be made to work in specific circumstances. The collaboration is described by Mobil and Los Alamos scientists in the companion article, "Toward Improved Prediction of Reservoir Flow Performance—Simulating Oil and Water Flows at the Pore Scale." The work has included the development of software to calculate relative permeabilities of oil and water in complex geometries. The software received an R&D-100 award for 1993.

The lattice-Boltzmann work is a superb example of applying the extraordinary computer power and expertise in computational physics available at Los Alamos to problems in petroleum production. The Laboratory's success at

projects of this type helped motivate the oil companies to ask that Congress substantially increase DOE support for oil and gas research and make those funds available for significant new collaborations between the oil and gas industry and the National Laboratories. The result is ACTI—the Advanced Computational Technology Initiative—an initiative that will receive \$30–50 million in funding in 1995.

In this article we will first sketch the basic ideas of the lattice-Boltzmann method and its general applicability. We will then explain how the lattice-gas and lattice-Boltzmann methods are adapted to describe the dynamics at the interfaces between two immiscible fluids such as oil and water and present some simulations of phase separation. Finally, we will mention a few of the new directions into which lattice-Boltzmann research is moving. Specific applications to the flows in oil reservoirs are discussed in the companion article.

Lattice Methods for Modeling Continuum Dynamics

Lattice methods, including the lattice-gas method and its derivative, the lattice-Boltzmann method, present powerful alternatives to the standard "top-down" and "bottom-up" approaches to modeling the behavior of physical systems. The "top-down" approach begins with a continuum description of macroscopic phenomena provided by partial differential equations. The Navier-Stokes equations for incompressible fluid flows are an example. Numerical techniques, such as finite-difference and finite-element methods, are then used to transform the continuum description into a discrete one in order to solve the equations numerically on a computer.

The "bottom-up" approach is based on the microscopic, particle description

provided by the equations of molecular dynamics; here the position and velocity of each atom or molecule in the system are closely followed by solving Newton's equations of motion. This microscopic description is straightforward to program on a computer but simulations using the largest computers presently available are limited to very small systems (ten million particles) and very short times (a few picoseconds). Therefore molecular dynamics simulations are more suitable for understanding the fundamental interactions that underlie macroscopic material properties than for modeling macroscopic dynamics.

Intermediate between the two schemes are the lattice-Boltzmann and lattice-gas methods, which might be considered mesoscopic approaches. The lattice methods begin from a particle description of matter: A gas of particles exists on a set of discrete points that are spaced at regular intervals to form a lattice. Time is also divided into discrete timesteps, and during each timestep particles jump to the next lattice site and then scatter according to simple kinetic rules that conserve mass, momentum, and energy. This simplified molecular dynamics is very carefully crafted to include the essentials of the real microscopic processes. Consequently the macroscopic, or averaged, properties of lattice simulations obey, to a good approximation, the desired continuum equations. Despite their origin in a particle description, lattice methods are essentially numerical schemes for studying averaged macroscopic behavior. They nevertheless retain the advantages of a particle description, including clear physical insight, easy implementation of boundary conditions, and fully parallel algorithms.

Because lattice methods are entirely local, they are extremely fast. A simulation of a simple lattice-gas model can

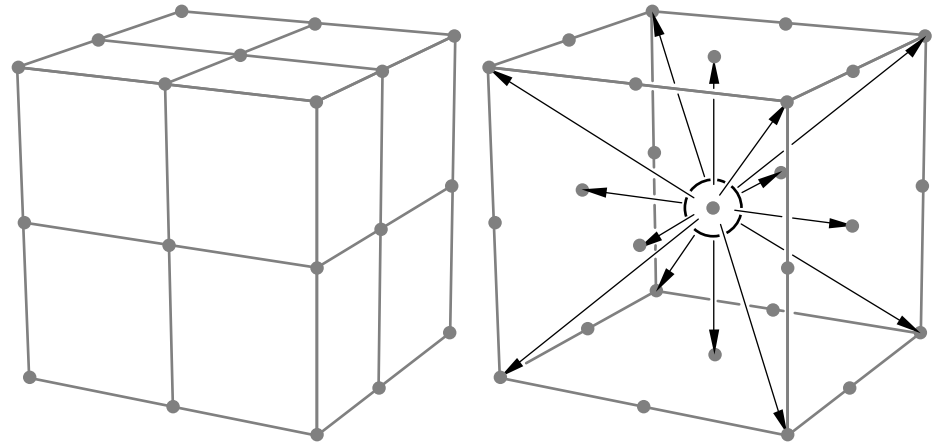


Figure 1. Allowed Velocities at a Lattice Site

The arrows indicate the magnitudes and directions of the allowed velocities e_i at a lattice site in a three-dimensional lattice-Boltzmann simulation. The lattice has a cubic structure. Six arrows point to nearest-neighbor sites. Eight arrows point along body diagonals. The sphere at the lattice site represents zero velocity, $e_0 = 0$. Lattice-Boltzmann simulations based on a proper equilibrium particle distribution and this minimum set of velocities preserve the desired isotropy of fluid properties.

attain a speed of 20 gigaflops on a 512-processor CM-5 Connection Machine. Typical simulations of flow through porous media include 100 million lattice sites and run for 5000 timesteps and therefore require only a few hours on that machine. When applied to periodic geometries, the three-dimensional lattice-Boltzmann model is able to achieve the same spatial resolution as conventional methods in half the time. Also, because boundary conditions—even complex ones—are imposed locally, lattice methods simulate flows in both simple and complex geometries with almost the same speed and efficiency. They are therefore suitable for simulating flows in the extremely complex geometries of porous media whereas conventional methods are not. Finally, developing code for lattice methods is considerably faster and easier than for traditional schemes.

General lattice models already exist for solving the equations of fluid flow, the wave equation, and the diffusion equation. Special versions of these models have been developed for simulating flow through porous media, turbulent flows, phase transitions, multiphase flows, chemically reacting flows, ther-

mohydrodynamics, magnetohydrodynamics, the dynamics of liquid crystals and the design of semiconductors. These models are validated by precise comparisons with other numerical algorithms, analytic results, and experiments.

The Lattice-Boltzmann Method

In the lattice-Boltzmann method, space is divided into a regular lattice (for example, a simple cubic lattice) and real numbers at each lattice site represent the single-particle distribution function at that site, which is equal to the expected number of identical particles in each of the available particle states i . In the simplest model, each particle state i is defined by a particle velocity, which is limited to a discrete set of allowed velocities. During each discrete timestep of the simulation, particles move, or hop, to the nearest lattice site along their direction of motion, where they “collide” with other particles that arrive at the same site. The outcome of the collision is determined by solving the kinetic (Boltzmann) equation for the new particle-distribution function at that site and the particle

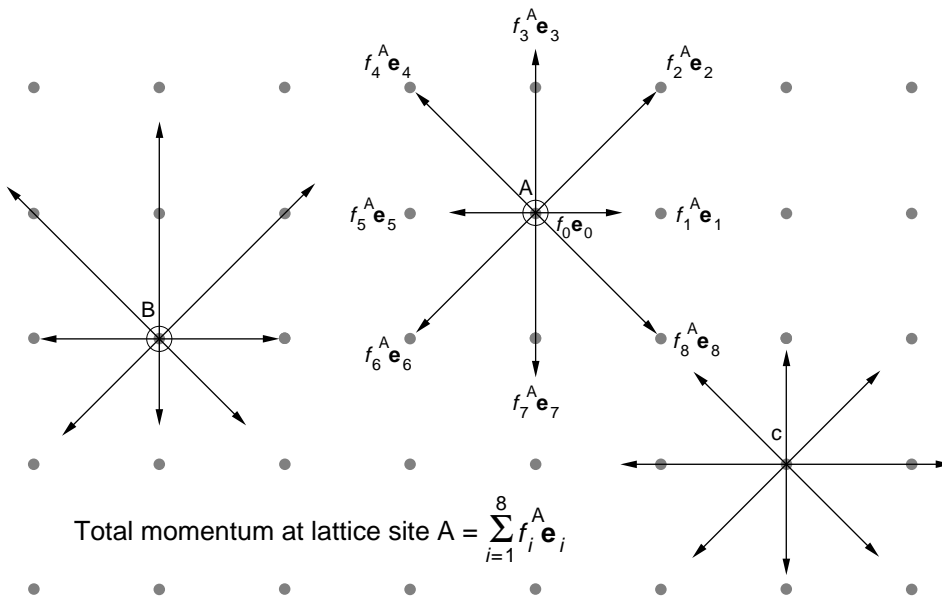


Figure 2. Momentum Distributions in Two Dimensions

The single-particle distribution function at each lattice site, $f_i(\mathbf{x}, t)$, equals the expected number of identical particles in each of the available particle states i . Particle velocities \mathbf{e}_i are limited to a discrete set defined by the geometry of the lattice, and the momentum distribution at a lattice site is equal to $f_i(\mathbf{x}, t)\mathbf{e}_i$. On a two-dimensional square lattice, eight directions of motion are possible, so $i = 0, 1, \dots, 8$. Each arrow indicates the momentum in one of the allowed directions of motion. The circles represent the rest particles. The figure shows several examples of momentum distributions in two dimensions. The distribution at site B has a net momentum in direction 3. The net momentum is equal to zero at sites A and C.

distribution function is updated.

More specifically, the single-particle distribution function at a single site in a simple cubic lattice is represented by a set of real numbers, $f_i(\mathbf{x}, t)$, the expected number of particles at lattice site \mathbf{x} and time t moving along the lattice vector \mathbf{e}_i , where each value of the index i specifies one of the allowed directions of motion. Figure 1 shows these directions. Six lattice vectors point to the six nearest-neighbor sites. Eight lattice vectors point along the body diagonals to the next sites along those diagonals. Thus particles can travel in fourteen directions from each lattice site. The ball in the center denotes the vector \mathbf{e}_0 , which is equal to 0 and represents particles that are not moving. In total, fifteen real numbers describe the particle distribution function at a site.

The first operation in each timestep Δt of the calculation is to advance the

particles to the next lattice site along their directions of motion. Since speed equals the distance traveled divided by Δt , this model has only three speeds, zero for particles at rest, c for particles moving to nearest-neighbor sites and $\sqrt{3}c$ for particles moving to the next sites along body diagonals. Usually the units are chosen such that the distance to nearest neighbors and Δt are unity, so that $c = 1$ and the lattice vector \mathbf{e}_i is numerically equal to the velocity of the particles moving in direction i . If we also set the mass of each particle equal to unity, the momentum in direction i at site \mathbf{x} and time t is just $f_i(\mathbf{x}, t)\mathbf{e}_i$. Figure 2 illustrates sample momentum distributions in two dimensions.

The second operation is to simulate particle collisions, which cause the particles at each lattice site to scatter into different directions. The collision rules are chosen to leave the sum of the

f_i 's unchanged. (No particles are lost.) The rules are also selected to conserve the total energy and momentum at each lattice site. To ensure that the particles have zero average velocity at boundaries (both perpendicular and parallel to the walls), one normally imposes "bounce-back" boundary conditions: Any flux of particles that hits a boundary simply reverses its velocity so that the average velocity at the boundary is automatically zero, as observed experimentally.

The outcome of collisions is very simply approximated by assuming that the momenta of the interacting particles will be redistributed at some constant rate toward an equilibrium distribution f_i^{eq} . This simplification is called the single-time-relaxation approximation. In mathematical terms, the time evolution of the single-particle distribution is given by

$$f_i(\mathbf{x} + \mathbf{e}_i, t + 1) = f_i(\mathbf{x}, t) - \frac{f_i(\mathbf{x}, t) - f_i^{\text{eq}}(\mathbf{x}, t)}{\tau},$$

where $i = 0, 1, \dots, 14$. The second term on the right-hand side is the simplified collision operator Ω_i . The rate of change toward equilibrium is $1/\tau$, the inverse of the relaxation time, and is chosen to produce the desired value of the fluid viscosity.

Figure 3 illustrates the single-time-relaxation approximation in two dimensions. The net momentum of the incoming state is zero. In that case, if no external forces exist, the equilibrium distribution consists simply of equal amounts of particle momentum in each of the allowed directions of motion. The collision rules will therefore force the larger f_i 's at the site to decrease and the smaller f_i 's to increase so that the particle distribution is closer to the equilibrium distribution after the collision than before. External forces (grav-

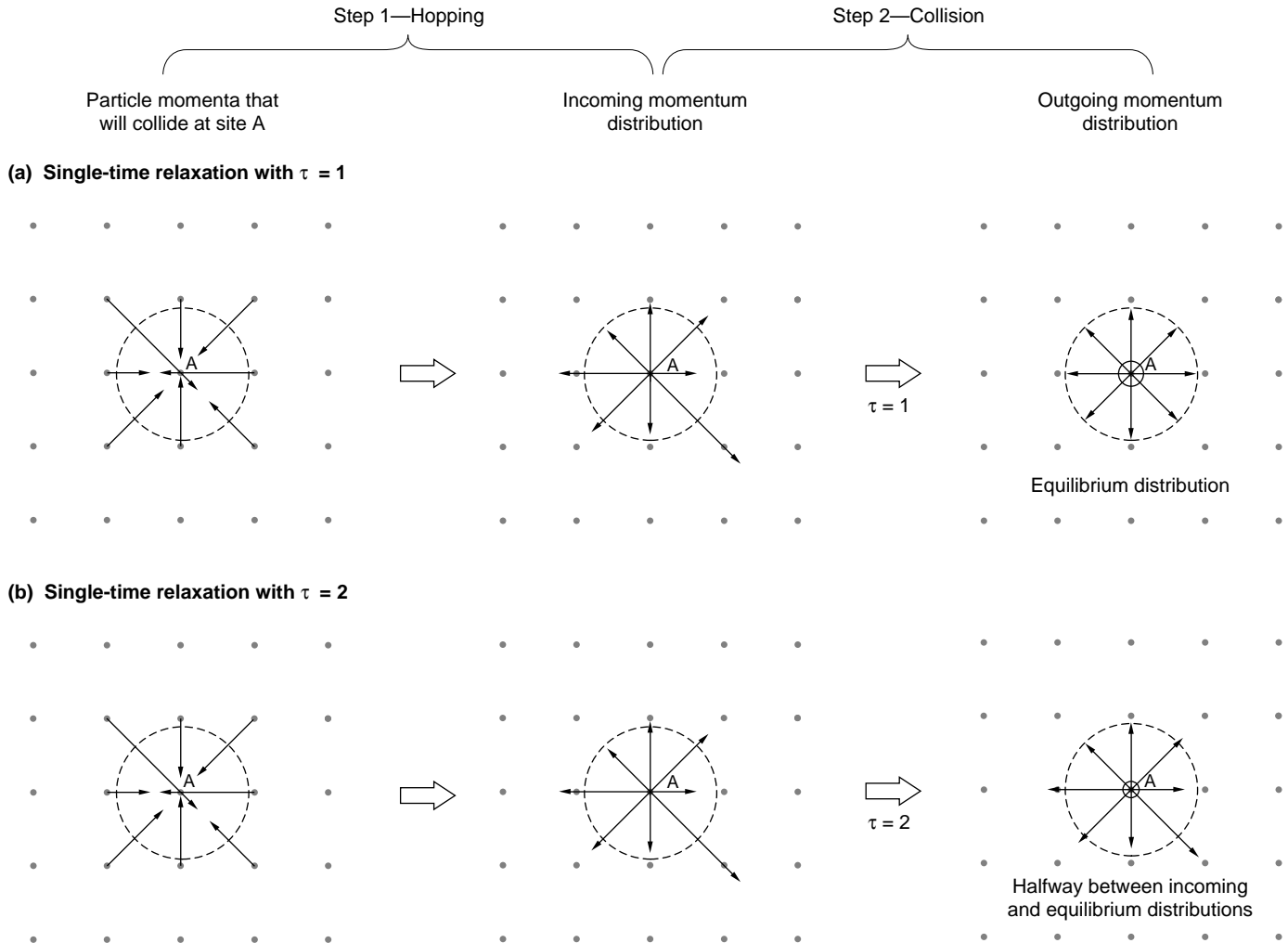


Figure 3. The Single-Time-Relaxation Process

In the single-time-relaxation approximation, the momentum distribution at each lattice site is forced toward the equilibrium distribution at each timestep. In the absence of external forces, the equilibrium distribution of a state with zero net momentum is just equal amounts of momentum in each direction. The figure illustrates the two calculational steps that occur during each timestep. First the incoming momentum distribution assembles at a lattice site as particles at the neighboring sites hop along their directions of motion to that site. Second, the incoming distribution changes, according to the single-time-relaxation collision rule, to an outgoing distribution that is closer to the equilibrium distribution. When $\tau = 1$, the incoming momentum distribution changes to the equilibrium distribution in one time step; when $\tau = 2$, the outgoing momentum distribution is halfway between the incoming and the equilibrium distributions.

ity or electromagnetic forces) can be added to the model and make the f_i 's grow in the direction of the net force and shrink in the opposite direction.

Although lattice models consist of a very simple set of rules, those rules lead to very complicated flow patterns. The opening pages of this article show a snapshot from a two-dimensional lattice-Boltzmann simulation performed by Lishi Luo here at the Laboratory. The simulation models the flow of air

past a rectangular plate. Periodic boundary conditions are imposed perpendicular to the plate; wind-tunnel boundary conditions (air flowing in from the left at higher pressure and out at the right at lower pressure) are imposed in the flow direction. Attached vortices and vortex shedding are evident, showing the complicated flow patterns that can be modeled with lattice-Boltzmann methods. Detailed, quantitative comparisons with other

methods and with experiment have verified the accuracy of this method. In general, lattice-Boltzmann simulations agree with exact solutions to the Navier-Stokes equations to second order in the lattice spacing and the timestep.

The lattice-Boltzmann method is an outgrowth of lattice-gas models, providing something akin to an ensemble average of many realizations of a lattice gas but without the unphysical effects.

Lattice-gas models differ from lattice-Boltzmann models in that the former follow the motion of actual particles whereas the latter utilize the expected values of the particle-distribution function. The gas is composed of identical particles that obey an exclusion principle: At most one particle can occupy a given particle state at a given time. Since $N_i(\mathbf{x}, t)$, the number of particles that occupy state i at a lattice site, is either 0 or 1, only single-digit binary arithmetic is required to update the particle numbers following each collision. Consequently lattice-gas calculations are extremely fast. Collision rules specify the possible outcomes of two-particle and three-particle collisions and the explicit choice at each site is implemented by random sampling. The lattice gas is therefore very noisy, and spatial and temporal averaging are required to obtain macroscopic quantities. Also the method has several major drawbacks, including restriction to low Reynolds number, lack of Galilean invariance (convective flow velocities don't add properly unless the system is at nearly constant density), and an unphysical equation of state in which the pressure depends on the local velocity in addition to the usual dependence on the local density.

In contrast, the lattice-Boltzmann method uses real (continuous) numbers to describe a particle distribution at each site; in this sense it approaches a continuum description. Although one pays a penalty in that floating-point arithmetic operations are now required to carry out the simulations, the continuum feature eliminates most of the noise associated with lattice-gas simulations. In addition, the local equilibrium particle distribution f_i^{eq} that appears in the collision operator can be chosen to eliminate the unphysical features of the lattice gas while including dependence on the local fluid variables only and

leading to the appropriate macroscopic equations. In other words, although lattice-Boltzmann simulations ignore particle-particle correlations and use simplified collision rules, the collision rules can be tailored to reproduce the correct evolution of the macroscopic behavior of a system in a wide variety of circumstances. The relevant equations are presented in the sidebar, "Equations of the Lattice-Boltzmann Method."

Strategies for Modeling the Flow of Two Fluid Phases

Modeling the flow of two immiscible fluids, such as oil and water, presents the difficulty of how to treat the dynamics at the interfaces between the two. These dynamics control phenomena such as the flow of oil and water through porous media, the development of viscous fingering, which occurs when water pushes oil, and dendrite formation (such as the growth of snowflakes). Traditional finite-difference and finite-element schemes can be used to model two-phase flow in simple flow geometries, but they are difficult to apply in the complicated geometries of porous media. Here we describe lattice methods for complex two-phase phenomena; these methods are being successfully applied to problems in oil recovery.

To extend lattice methods to the flow of two immiscible fluids (say red and blue fluids), one must double the information at each lattice site. In the lattice-Boltzmann approach this is done by postulating a single-particle distribution for each of the fluids. It is also necessary to create new scattering rules that will cause the two fluids to separate at an interface and to emulate the effects of surface tension. The essential idea is to compute a force at each lattice site that depends on, say, the red-

fluid density at the neighboring sites and that pushes the red fluid in the direction of increasing red-fluid density. The effect of the force must be added to the collision operator in such a way that it does not change the total momentum of the two fluids. The size of the force must be chosen to produce the desired surface tension between the two fluids. Under these rules, an initially random mixture of the red and blue fluids should unmix and form an equilibrium distribution in which a spherical interface separates the two fluids, as expected physically in mixtures of oil and water.

Lattice-gas simulations of immiscible fluids. The extension of lattice methods to two immiscible fluids was done originally in the context of lattice gases. Our particular method implements the effects of surface tension in a purely local manner—through the introduction of two species of colored holes. A hole acts like a massless "ghost" particle that moves freely on the lattice but carries no momentum. A hole of one color is created at a lattice site when an incoming particle of that color is scattered into a new direction; the new hole moves in the incoming direction of the scattered particle and thereby carries a memory of that particle. The hole is annihilated, or disappears, when it meets a particle of the same color traveling in the same direction.

To create surface tension, a new collision rule is added to the usual ones that causes the colored particles at a site to respond to the presence of colored holes by moving in the opposite direction. The overall effect is like that of a short-range attractive potential among particles of the same color. Although the "color potential" has a range of more than one lattice spacing, only local information is needed to compute the results of collision at each lattice

site; consequently the colored-hole scheme is much faster for modeling the effects of surface tension than methods requiring the calculation of nearest-neighbor color gradients.

The colored-hole scheme has been used successfully to simulate phase separation, surface tension between phases, and contact angles between fluids and solid walls. Figure 4 shows the results of a lattice-gas simulation of red and blue fluids: Red drops have formed in blue fluid starting from a random initial configuration in which the density of red particles was equal to one-fifth the density of blue particles. Figure 5 demonstrates that the results shown in Figure 4 obey the Laplace formula for surface tension. That is, the pressure drop Δp across the boundary of each drop is given by

$$\Delta p = p_{\text{red}} - p_{\text{blue}} = \frac{\sigma}{R},$$

where p_{red} is the pressure of red particles inside the drop, p_{blue} is the pressure of blue particles outside the drop, σ is the surface-tension coefficient, and R is the radius of the drop. Figure 5a shows the pressure versus the distance r from the center of a drop; the pressure change at the red-blue interface demonstrates the existence of surface tension. Figure 5b shows Δp , the pressure change across the interface of the drop for drops of different radii R . As required by the Laplace formula, the pressure change is directly proportional to $1/R$.

In addition to surface tension at fluid-fluid interfaces, the two-fluid lattice-gas model must also simulate interactions between the fluids and the walls. Typically, when the interface between two fluids intersects a solid wall, the angle of intersection is fixed. Figure 6 illustrates the two contact an-

gles, one for each fluid, formed at the point of intersection. By definition, the sum of the two contact angles is equal to 180° .

The contact angle of a fluid decreases as the affinity, or preference, of the wall for that fluid increases. That preference is called wettability. On a strongly water-wet surface, water droplets in a background of oil will tend to spread out on the surface and the contact angle of water will be close to 0° . On a strongly oil-wet surface, those water droplets will bead up and the contact angle will be close to 180° . In our lattice-gas simulations, wettability is controlled by a special rule for collisions of colored holes with the wall: With a certain probability P , a hole of either color bounces back and becomes, say, a red hole. This rule can produce contact angles for the red fluid of between 20° and 180° .

Figure 7 shows the results of colored-hole two-fluid simulations in which different values of the probability P for the upper and lower surfaces of a wall (white) have led to marked differences in the contact angles of the red fluid. The results shown were obtained by averaging 12 realizations of the lattice gas.

Lattice-Boltzmann model for two fluids. Lattice-gas simulations produce realistic surface phenomena for immiscible fluids, but they do so only when temporal and spatial averaging are used to eliminate the noise induced by particle fluctuations. A more convenient approach is to extend the lattice-Boltzmann method to colored fluids and use color gradients, or nearest-neighbor interactions, for modeling interfacial dynamics. In this approach, each colored fluid is described by a single-particle distribution function, and each is assigned a number density as well as a characteristic relaxation time, which

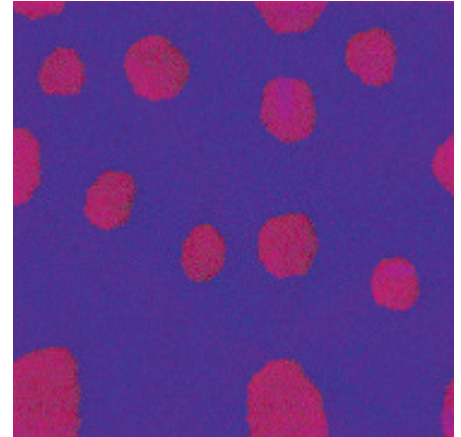


Figure 4. Droplet Formation in Two-Phase Lattice-Gas Mixtures

The figure shows the results of a lattice-gas simulation in which the dynamics of the interface between red and blue phases was modeled by using the colored-hole scheme summarized in the text. The simulation produced this distribution of red droplets in a blue background after 17,200 time steps, starting from a random initial distribution of red and blue particles on 512×512 lattice sites. The total particle density is 0.4 and the ratio of red to blue particles is 1/5. Because the lattice gas is very noisy, the droplets have very rough surfaces, in contrast to the smooth surfaces expected at fluid-fluid interfaces. Consequently the results of the lattice-gas simulations must be averaged spatially and temporally to be compared with the macroscopic description of immiscible fluids (see Figure 5). This figure should also be compared with Figure 8, which shows that the lattice-Boltzmann method produces smoother, more realistic droplets.

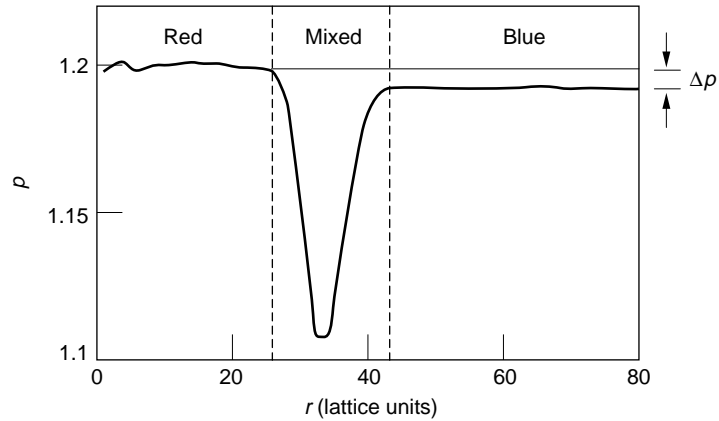
Figure 5. Verification of the Laplace Formula for Surface Tension in Two-Phase Lattice-Gas Simulations

For real droplets in a mixture of immiscible fluids, a pressure difference across the interface balances the surface tension created by the mutual attraction of particles of like kind. The relation is given by the Laplace formula, $\Delta p = \sigma/R$ where σ is the surface-tension coefficient and R is the radius of the droplet. To compare the lattice-gas results of Figure 4 with that macroscopic description, the pressure is calculated from the local particle-number density, $p = n/3$, and the results are averaged temporally over 1000 timesteps and spatially in the circumferential direction around the center of each droplet.

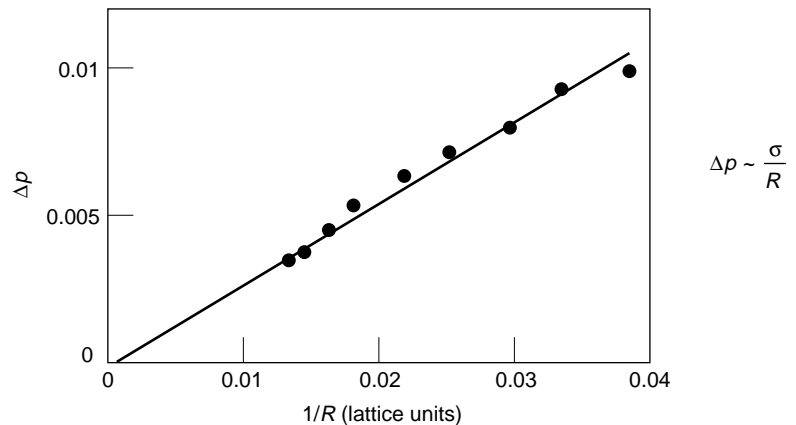
(a) That average pressure is plotted as a function of the distance r from the center of a droplet. The pressure difference between the red phase and the blue phase demonstrates the existence of surface tension in these two-phase simulations.

(b) The pressure difference across the droplet surface is plotted as a function of $1/R$, where R is the radius of the droplet. The linear relation shows that the surface tension coefficient σ is a constant, as required by the Laplace formula. Thus the averaged results of the colored-hole, two-phase lattice-gas simulations reproduce the correct macroscopic interfacial dynamics.

(a) Pressure profile of red droplet



(b) Pressure difference across red-blue interface for red droplets of different radii (R)



determines its viscosity. The collision operator now has two parts. The first part of Ω_i induces single-time relaxation to equilibrium, as was done for the single phase. Again each fluid tends to relax to a local equilibrium distribution that depends on the local density and velocity and is analogous to the single-phase equilibrium distribution presented in the sidebar. The mass of each fluid is conserved and the total momentum of the system is conserved.

The second part of the collision operator Ω_i depends on the red and blue densities at nearest neighbors and deter-

mines the dynamics at the fluid-fluid interfaces. It is given by

$$\frac{A_k}{2} |\mathbf{F}| \left[\frac{(\mathbf{e}_i \cdot \mathbf{F})^2}{|\mathbf{F}|^2} - 1/2 \right],$$

where k denotes red or blue, A_k is a free parameter that controls the surface tension, and \mathbf{F} is the local color gradient, defined as

$$\mathbf{F}(\mathbf{x}) = \sum_i \mathbf{e}_i [\rho_r(\mathbf{x} + \mathbf{e}_i) - \rho_b(\mathbf{x} + \mathbf{e}_i)].$$

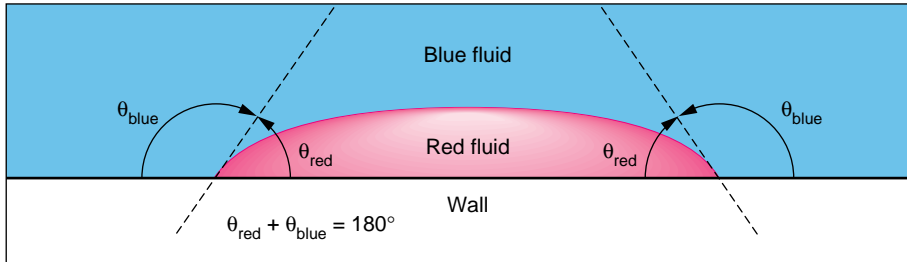


Figure 6 . The Definition of Contact Angle

The contact angle is defined as the angle between a two-fluid interface and a solid surface. As shown in the figure, each fluid has its own contact angle and the sum of the two must equal 180° . The wetting fluid (the fluid that tends to wet the surface) has a contact angle of less than 90° , and the nonwetting fluid (the fluid that has less affinity for the solid surface) has a contact angle of greater than 90° .

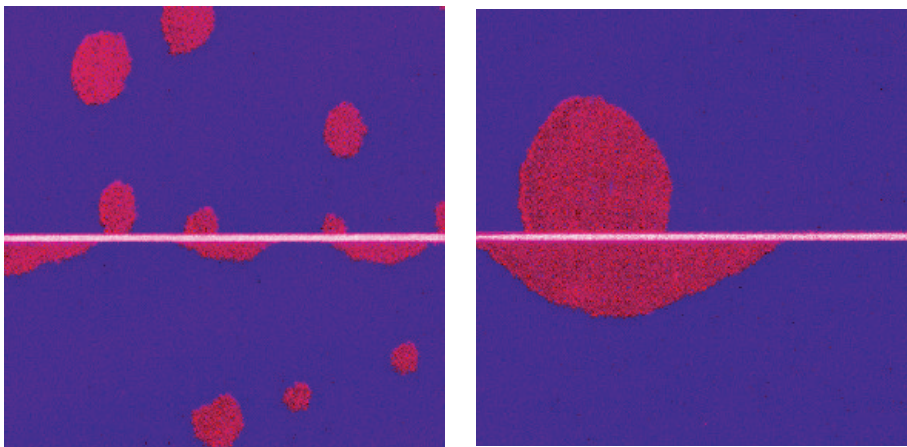


Figure 7. Contact Angles in Two-Phase Lattice-Gas Simulations

This simulation demonstrates the ability of lattice-gas simulations to model different contact angles, or different wettability conditions, as required in studies of porous-media flow. The simulation conditions are identical to those described in Figure 4 except that a solid horizontal wall (white) divides the computational box into an upper and a lower region. The upper surface of the wall has a greater affinity for the blue fluid than for the red fluid, whereas the opposite is true of the lower surface of the wall. The affinity, or preference, of the wall for one fluid over the other is modeled by a special rule for collisions between a hole and the wall that controls the color of the hole flux at the walls. The rule specifies that when a hole of either color collides with a wall, there is a probability P that it will bounce back and become a red hole. In the simulation shown the upper surface has $P = 0.2$; that choice produces a surface that is strongly blue-wet, and the contact angle of the red fluid is greater than 90° . The lower surface has $P = 0.8$, which results in a strongly red-wet surface and a contact angle for the red fluid of close to 0° .

In the incompressible limit, the density in a single-phase region is uniform and therefore the local color gradient is zero. Thus this second part of the collision operator does not contribute. In a mixed-phase region, this part of the collision operator maintains the interfaces between fluids by forcing the momentum of the red fluid, $\mathbf{j}^r = \sum_i f_i^r \mathbf{e}_i$, to align with the direction of the local color gradient. In other words, the red density at an interface is redistributed to maximize the quantity $-(\mathbf{j} \cdot \mathbf{F})$. The blue-particle distribution can then be obtained by applying mass conservation along each direction: $f_i^b = f_i - f_i^r$.

Using a scaling and expansion procedure similar to that used to derive the macroscopic behavior of a single phase (see sidebar), one can rigorously prove that each fluid will obey the Navier-Stokes equations and that the pressure difference across fluid-fluid interfaces will obey the Laplace formula. Results of two-phase lattice-Boltzmann simulations have also demonstrated that the surface-tension coefficient σ is a constant independent of the radius of a drop.

Over the past three years, several Los Alamos scientists and several scientists from Mobil Oil Company have developed a collaboration, applying the two-phase model to study flows through porous media. The results of simulations in two and three dimensions are shown in the companion article. The goal of the collaboration is to develop a greater understanding of the fundamental physics related to enhanced oil recovery, and to predict relative permeabilities and other bulk properties of porous media from basic properties of the fluid-fluid and fluid-rock interactions. So far, lattice-Boltzmann simulations look promising: Initial results compare well with experimental measurements of flow patterns and relative permeabilities.

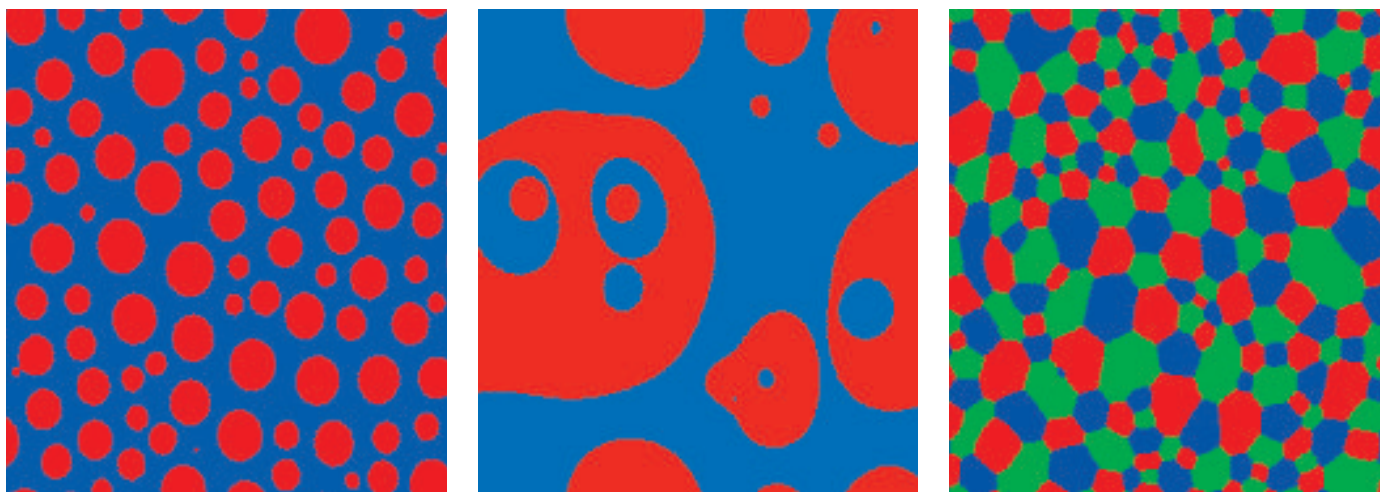


Figure 8. Lattice-Boltzmann Simulations of Phase Separation

Each of the three figures is a snapshot of a lattice-Boltzmann simulation after 6000 timesteps, starting from an initial random distribution of the phases present. (a) The red and blue densities are 0.4 and 0.6, respectively. Since the density of the red fluid is lower than that of the blue, red drops form in a blue background. At each site the lattice-Boltzmann simulation produces, in essence, an ensemble average of lattice-gas simulations, so the boundaries between the red and blue fluids are much smoother in this simulation than in the corresponding lattice-gas simulation shown in Figure 4. (b) The densities of the red and blue fluids are 0.5 and 0.5. Phase separation again takes place readily but produces a different pattern than in (a) because the densities of the two phases are equal. (c) The model for immiscible fluids can be applied to any number of phases. Here there are three phases, red, blue, and green, and each has a density equal to 0.33. Again the three phases separate from an initial random distribution.

Applications and Extensions

Lattice multiphase models have been used for investigating the details of interface dynamics, including the well-known phenomena Rayleigh-Taylor instability, Saffman-Taylor instability, and domain growth in the separation of two immiscible fluids (see Figure 8). The two-phase lattice-Boltzmann model has been extended to three and four phases to study critical and off-critical quenches and phase transitions.

Features of interest in biological problems have also been added to lattice-Boltzmann models. For example, the addition of a surfactant fluid to two-phase flows is being used to study the formation of lipid bilayers. If one end of the surfactant prefers the red fluid and the other end of the surfactant prefers the blue fluid, the fluids are

forced to have a much more extensive interface. This approach is being used for studying the self-assembly of biological membranes, surfactant effects in two- and three-dimensional multiphase models, the formation of membranes, and self-assembly of amphiphilic structures at aqueous organic interfaces.

Simple reaction-diffusion systems are also being modeled with lattice-Boltzmann methods. Recent work has shown that these simple systems can exhibit extremely complicated and unexpected behavior. For example, solutions have been found that behave like biological cells; that is, concentrations of chemical densities within well-defined regions of the solution divide as cells divide.

The lattice-Boltzmann method, like any new method, is going through an exploratory stage to determine its limitations and optimal areas of application.

Recently, stability analyses have been completed, and the conditions under which the method becomes unstable (as do all finite-difference schemes) have been determined. When the allowed number of speeds is increased from three to many, a thermohydrodynamic version of the lattice-Boltzmann method emerges. Such a model has been developed and tested by several scientists. The current lattice models for reacting systems are based on simple isothermal models, but they can easily be extended to include temperature effects and to model phase transitions in which the reaction rate depends on local temperature.

Clearly, the lattice-Boltzmann method is still in the developmental stages. Because it is relatively straightforward to introduce new physics into the model, we expect it will be applied to many additional physical phenomena. ■

Acknowledgements

Members of the Lattice-Boltzmann Research Effort: Francis Alexander, Mario Ancona, Daryl Grunau, Brosl Hasslacher, Shuling Hou, Nathan Kreisberg, Turab Lookman, Lishi Luo, Daniel Martinez, Guy R. McNamara, Balu Nadiga, Silvina Ponce-Dawson, Xiaowen Shan, James Sterling, and Qisu Zou.

Further Reading

U. Frisch, B. Hasslacher, and Y. Pomeau. 1986. Lattice-gas automata for the Navier-Stokes equation. *Physical Review Letters* 56: 1505.

S. Wolfram. 1986. Cellular automata fluids I: Basic theory. *Journal of Statistical Physics* 45: 471–526.

G. McNamara and G. Zanetti. 1988. Use of the Boltzmann equation to simulate lattice-gas automata. *Physical Review Letters* 61: 2332.

D. H. Rothman and J. M. Keller. 1988. Immiscible cellular-automaton fluids. *Journal of Statistical Physics* 52: 1119.

S. Succi, P. Santangelo, and R. Benzi. 1988. High-resolution lattice-gas simulation of two-dimensional turbulence. *Physical Review Letters* 61: 2738–2740.

S. Chen, H. Chen, D. Martínez, and W. H. Matthaeus. 1991. Lattice Boltzmann model for simulation of magnetohydrodynamics. *Physical Review Letters* 67: 3776.

S. Chen, G. D. Doolen, K. Eggert, D. Grunau, and E. Y. Loh. 1991. Local lattice-gas model for immiscible fluids. *Physical Review A* 43: 7053.

A. K. Gunstensen, D. H. Rothman, S. Zaleski, and G. Zanetti. 1991. Lattice Boltzmann model for immiscible fluids. *Physical Review A* 43: 4320.

Y. H. Qian, D. d'Humieres, and P. Lallemand. 1992. Lattice BGK models for Navier-Stokes equation. *Europhysics Letters* 17: 479.

F. J. Alexander, S. Chen, and D. W. Grunau. 1993. Hydrodynamic spinodal decomposition: Growth kinetics and scaling functions. *Physical Review B* 48: 634–637.

F. J. Alexander, S. Chen, and J. D. Sterling. 1993. Lattice-Boltzmann thermohydrodynamics. *Physical Review E* 47: 2249–2252.



Seated at the monitor, Shiyi Chen; standing, left to right, Gary D. Doolen, Kenneth G. Eggert, and Wendy E. Soll. [Soll is a co-author of “Toward Improved Prediction of Reservoir Flow Performance.”]

Shiyi Chen first joined the Laboratory as a postdoctoral fellow in 1987; he was an Oppenheimer Fellow from 1990 to 1992. In 1992 he became a staff member in the Complex Systems Group of the Theoretical Division. His research interests include lattice-gas automata and lattice-Boltzmann computational methods, the statistical theory of fluid turbulence, and two-phase and compressible fluid flows. A native of the People’s Republic of China, Chen earned his B.S. in 1981 from Zhejiang University and his M.S. and Ph.D. from Beijing University in 1984 and 1987.

Gary D. Doolen joined the Laboratory’s Applied Theoretical Physics Group in 1975. His research interests include nuclear physics, plasma physics, magnetohydrodynamics, neural networks, and nonlinear mathematics. Doolen was with the Center for Nonlinear Studies from 1988 through 1993. He is now the leader of the Complex Systems Group. He received his B.S. in engineering science in 1961 and his M.S. and Ph.D. in physics in 1964 and 1967 from Purdue University.

Kenneth G. Eggert joined the Laboratory’s Geoanalysis Group as a section leader in 1987 and became group leader in 1992. His research interests include use of simulation in the management of surface and subsurface hydrologic systems, physics of multiphase flow, and parallel computation in the earth sciences. From 1984 to 1987 he was a physicist in the Earth Sciences Department at Lawrence Livermore Laboratory where he participated in the Yucca Mountain Project. Eggert received his B.S. in aerospace engineering from Purdue University in 1969 and his M.S. and Ph.D. in civil engineering from Colorado State University at Fort Collins in 1976 and 1980. Eggert is a member of the Society of Petroleum Engineers and the American Geophysical Union.

R. Benzi, S. Succi, and M. Vergassola. 1993. The lattice Boltzmann equation: Theory and applications. *Physics Reports* 222: 145–197.

G. D. Doolen, editor. 1993. *Lattice Gas Methods for PDE’s: Theory, Application and Hardware*. *Physica D* 47.

D. Grunau, S. Chen, and K. Eggert. 1993. A lattice Boltzmann model for multi-phase fluid flows. *Physics of Fluids A* 5: 2557.

W. E. Soll, S. Y. Chen, K. G. Eggert, D. W. Grunau, and D. R. Janecky. July, 1994. Application of the lattice-Boltzmann/lattice gas technique to multi-fluid flow in porous media. In *Proceedings of Computational Methods in Water Resources*.

Equations of the Lattice-Boltzmann Method

The Boltzmann equation for any lattice model is an equation for the time evolution of $f_i(\mathbf{x}, t)$, the single-particle distribution at lattice site \mathbf{x} :

$$f_i(\mathbf{x} + \mathbf{e}_i, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i(f(\mathbf{x}, t)),$$

where $\Omega_i = \Omega_i(f(\mathbf{x}, t))$ is the local collision operator at that site, $i = 0, 1, \dots, 14$, and Δt is assigned a value of unity. Since the usual aim of lattice methods is to model macroscopic dynamics, the “exact” collision operator is unnecessarily complex and therefore numerically inefficient. Two groups (see Chen *et al.* and Qian *et al.* in the Further Reading) nearly simultaneously suggested that the collision operator be approximated by a single-time-relaxation process in which relaxation to some appropriately chosen equilibrium distribution occurs at some constant rate. In particular the collision term, $\Omega(f)$, is replaced by the single-time-relaxation approximation,

$$\Omega_i(f(\mathbf{x}, t)) = -\frac{f_i(\mathbf{x}, t) - f_i^{\text{eq}}(\mathbf{x}, t)}{\tau}.$$

The appropriately chosen equilibrium distribution, denoted by f^{eq} , depends on the local fluid variables, and $1/\tau$ is the rate of approach to this equilibrium. The relations $\sum_i \Omega_i = 0$ and $\sum_i \mathbf{e}_i \Omega_i = 0$ must be true to conserve mass and momentum, respectively. In order for the fluid to have Galilean-invariant convection and a pressure that does not depend on velocity, an appropriate equilibrium distribution, f_i^{eq} , must be assumed. For a two-dimensional hexagonal lattice, the formula is:

$$f_i^{\text{eq}} = \frac{\rho(1 - \alpha)}{6} + \frac{\rho}{3} \mathbf{e}_i \cdot \mathbf{v} + \frac{2\rho}{3} (\mathbf{e}_i \cdot \mathbf{v})^2 - \frac{\rho}{6} \mathbf{v}^2$$

and

$$f_0^{\text{eq}} = \alpha\rho - \rho\mathbf{v}^2.$$

(The corresponding formulas for the cubic lattice appear in the article by Alexander, Chen, and Grunau listed in the Further Reading.) In these equations the density $\rho(\mathbf{x}, t) = mn(\mathbf{x}, t)$ (where m is the mass of each particle), the number density $n(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t)$, and α is a free parameter related to the sound speed as shown below. For the lattice-Boltzmann method, the particle distribution does not have an upper bound. The only requirement is that $f_i \geq 0$.

To derive the macroscopic equations obeyed by this model, one performs a Taylor expansion in time and space and takes the long-wavelength and low-frequency limit of the lattice-Boltzmann equation for the single-particle distribution. The result is a continuum form of the Boltzmann equation correct to second order in the lattice spacing and the timestep. A scaling expansion argument, the assumption of single-time relaxation, and the neglect of higher-order terms lead to the following final form of the macroscopic equations obeyed by the simulated system

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v_\beta)}{\partial x_\beta} = 0, \text{ the equation of mass continuity;}$$

$$\frac{\rho \partial v_\alpha}{\partial t} + \rho v_\beta \frac{\partial v_\alpha}{\partial x_\beta} = - \frac{\partial p}{\partial x_\alpha} + \frac{\partial}{\partial x_\beta} \left[\frac{\lambda}{\rho} \left(\frac{\partial \rho v_\gamma}{\partial x_\gamma} \delta_{\alpha\beta} + v_\alpha \frac{\partial \rho}{\partial x_\beta} + v_\beta \frac{\partial \rho}{\partial x_\alpha} \right) \right] + \frac{\partial}{\partial x_\beta} \left[\mu \left(\frac{\partial v_\beta}{\partial x_\alpha} + \frac{\partial v_\alpha}{\partial x_\beta} \right) \right],$$

the equation of momentum conservation; and

$$p = \frac{1 - \xi}{2} \rho.$$

the equation of state.

In the above equations, v_β is the component of the velocity in the β -direction; p is the pressure; and the sound speed, c_s , is $\sqrt{(1 - \xi)/2}$, where ξ is a free parameter. The shear viscosity, μ , and the bulk viscosity, λ , are given by

$$\mu = \frac{2\tau - 1}{8} \rho$$

and

$$\lambda = \frac{(\tau - 1/2)(2\xi - 1)}{4} \rho.$$

The above equations converge to the exact incompressible Navier-Stokes equations only when the derivatives of the number density in the second viscosity term on the right-hand side of the equation are small. Since the gradients of the density are $O(v^2)$, the unphysical terms in the momentum-conservation equation are correct to order (v^3) . Thus, although the physics of the lattice-Boltzmann method contains compressibility effects, one may come arbitrarily close to solving the incompressible Navier-Stokes equations by reducing the Mach number (through the choice of α) and thereby reducing the simulated flow to very low speed. (Nevertheless the compressibility effects in the lattice-Boltzmann approach are physical and the method can also be used to simulate compressible fluids.)

Traditional methods for solving incompressible flows, such as finite-difference or finite-element, require solution of a Poisson equation for the pressure term, which is induced by the mass-continuity equation and the momentum-conservation equation. In the lattice-Boltzmann approach, this time-consuming step is avoided because the incompressibility requirement has been relaxed and the effects of pressure changes are controlled by an equation of state rather than a Poisson equation. It can be argued that the conventional methods most closely related to the lattice-Boltzmann method are the pseudocompressible algorithms for solving incompressible fluid flows. \square

Equations of the Lattice-Boltzmann Method

The Boltzmann equation for any lattice model is an equation for the time evolution of $f_i(\mathbf{x}, t)$, the single-particle distribution at lattice site \mathbf{x} :

$$f_i(\mathbf{x} + \mathbf{e}_i, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i(f(\mathbf{x}, t)),$$

where $\Omega_i = \Omega_i(f(\mathbf{x}, t))$ is the local collision operator at that site, $i = 0, 1, \dots, 14$, and Δt is assigned a value of unity. Since the usual aim of lattice methods is to model macroscopic dynamics, the “exact” collision operator is unnecessarily complex and therefore numerically inefficient. Two groups (see Chen *et al.* and Qian *et al.* in the Further Reading) nearly simultaneously suggested that the collision operator be approximated by a single-time-relaxation process in which relaxation to some appropriately chosen equilibrium distribution occurs at some constant rate. In particular the collision term, $\Omega(f)$, is replaced by the single-time-relaxation approximation,

$$\Omega_i(f(\mathbf{x}, t)) = -\frac{f_i(\mathbf{x}, t) - f_i^{\text{eq}}(\mathbf{x}, t)}{\tau}.$$

The appropriately chosen equilibrium distribution, denoted by f^{eq} , depends on the local fluid variables, and $1/\tau$ is the rate of approach to this equilibrium. The relations $\sum_i \Omega_i = 0$ and $\sum_i \mathbf{e}_i \Omega_i = 0$ must be true to conserve mass and momentum, respectively. In order for the fluid to have Galilean-invariant convection and a pressure that does not depend on velocity, an appropriate equilibrium distribution, f_i^{eq} , must be assumed. For a two-dimensional hexagonal lattice, the formula is:

$$f_i^{\text{eq}} = \frac{\rho(1 - \alpha)}{6} + \frac{\rho}{3} \mathbf{e}_i \cdot \mathbf{v} + \frac{2\rho}{3} (\mathbf{e}_i \cdot \mathbf{v})^2 - \frac{\rho}{6} \mathbf{v}^2$$

and

$$f_0^{\text{eq}} = \alpha\rho - \rho\mathbf{v}^2.$$

(The corresponding formulas for the cubic lattice appear in the article by Alexander, Chen, and Grunau listed in the Further Reading.) In these equations the density $\rho(\mathbf{x}, t) = mn(\mathbf{x}, t)$ (where m is the mass of each particle), the number density $n(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t)$, and α is a free parameter related to the sound speed as shown below. For the lattice-Boltzmann method, the particle distribution does not have an upper bound. The only requirement is that $f_i \geq 0$.

To derive the macroscopic equations obeyed by this model, one performs a Taylor expansion in time and space and takes the long-wavelength and low-frequency limit of the lattice-Boltzmann equation for the single-particle distribution. The result is a continuum form of the Boltzmann equation correct to second order in the lattice spacing and the timestep. A scaling expansion argument, the assumption of single-time relaxation, and the neglect of higher-order terms lead to the following final form of the macroscopic equations obeyed by the simulated system

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v_\beta)}{\partial x_\beta} = 0, \text{ the equation of mass continuity;}$$

$$\frac{\rho \partial v_\alpha}{\partial t} + \rho v_\beta \frac{\partial v_\alpha}{\partial x_\beta} = - \frac{\partial p}{\partial x_\alpha} + \frac{\partial}{\partial x_\beta} \left[\frac{\lambda}{\rho} \left(\frac{\partial \rho v_\gamma}{\partial x_\gamma} \delta_{\alpha\beta} + v_\alpha \frac{\partial \rho}{\partial x_\beta} + v_\beta \frac{\partial \rho}{\partial x_\alpha} \right) \right] + \frac{\partial}{\partial x_\beta} \left[\mu \left(\frac{\partial v_\beta}{\partial x_\alpha} + \frac{\partial v_\alpha}{\partial x_\beta} \right) \right],$$

the equation of momentum conservation; and

$$p = \frac{1 - \xi}{2} \rho.$$

the equation of state.

In the above equations, v_β is the component of the velocity in the β -direction; p is the pressure; and the sound speed, c_s , is $\sqrt{(1 - \xi)/2}$, where ξ is a free parameter. The shear viscosity, μ , and the bulk viscosity, λ , are given by

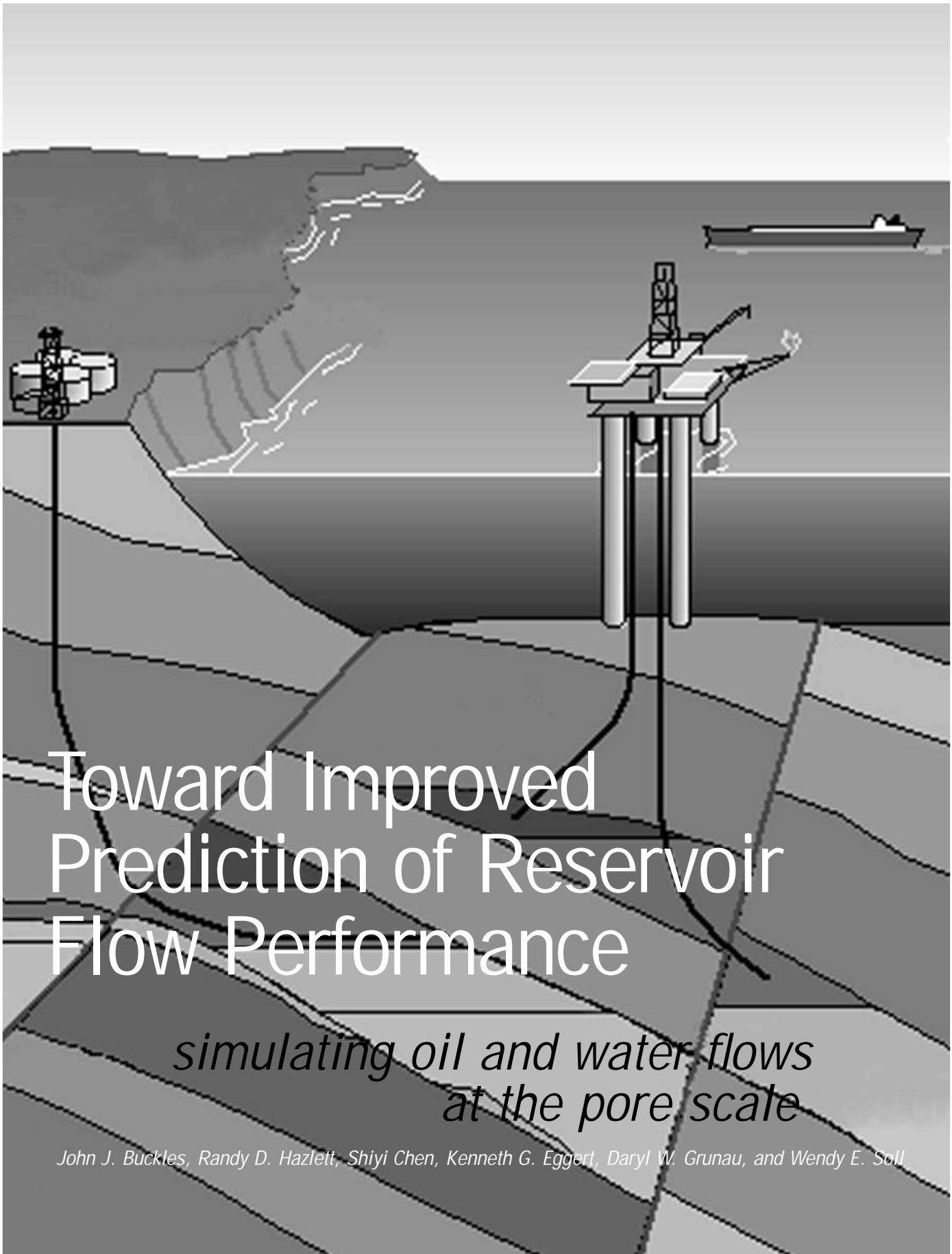
$$\mu = \frac{2\tau - 1}{8} \rho$$

and

$$\lambda = \frac{(\tau - 1/2)(2\xi - 1)}{4} \rho.$$

The above equations converge to the exact incompressible Navier-Stokes equations only when the derivatives of the number density in the second viscosity term on the right-hand side of the equation are small. Since the gradients of the density are $O(v^2)$, the unphysical terms in the momentum-conservation equation are correct to order (v^3) . Thus, although the physics of the lattice-Boltzmann method contains compressibility effects, one may come arbitrarily close to solving the incompressible Navier-Stokes equations by reducing the Mach number (through the choice of α) and thereby reducing the simulated flow to very low speed. (Nevertheless the compressibility effects in the lattice-Boltzmann approach are physical and the method can also be used to simulate compressible fluids.)

Traditional methods for solving incompressible flows, such as finite-difference or finite-element, require solution of a Poisson equation for the pressure term, which is induced by the mass-continuity equation and the momentum-conservation equation. In the lattice-Boltzmann approach, this time-consuming step is avoided because the incompressibility requirement has been relaxed and the effects of pressure changes are controlled by an equation of state rather than a Poisson equation. It can be argued that the conventional methods most closely related to the lattice-Boltzmann method are the pseudocompressible algorithms for solving incompressible fluid flows. \square



Toward Improved Prediction of Reservoir Flow Performance

*simulating oil and water flows
at the pore scale*

John J. Buckles, Randy D. Hazlett, Shiyi Chen, Kenneth G. Eggert, Daryl W. Grunau, and Wendy E. Soll

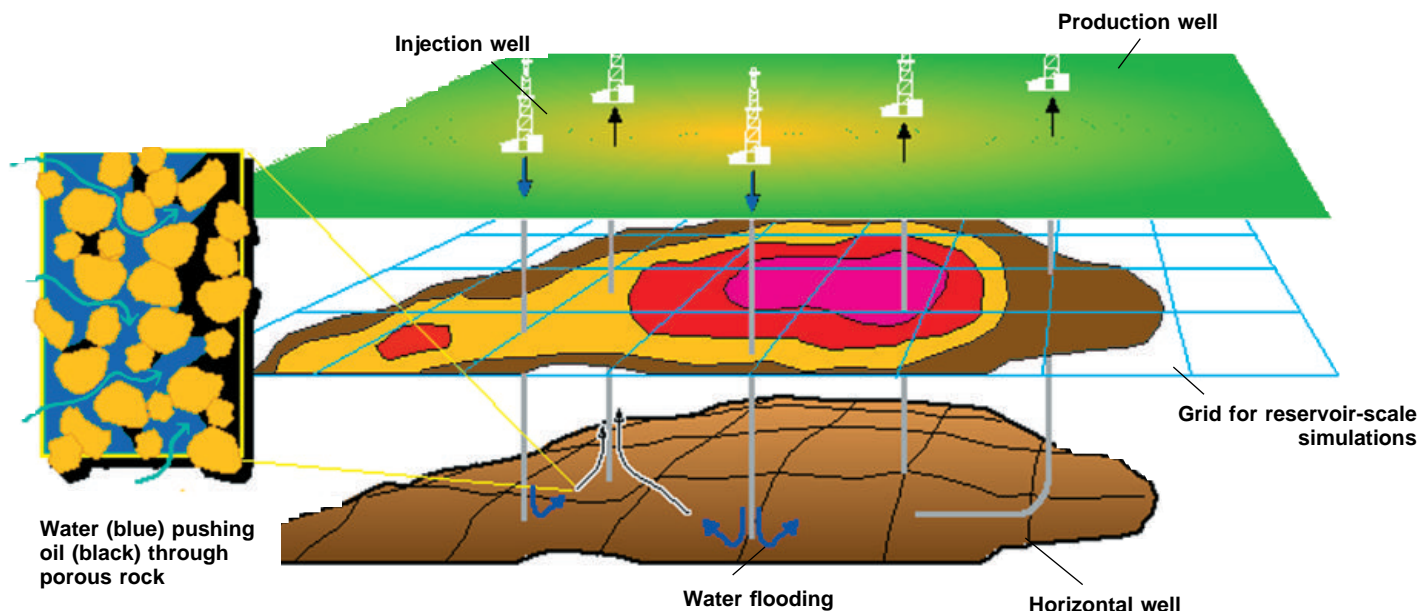


Figure 1. A Modern Oil-Recovery Project

This illustration of a modern oil-recovery project shows both production wells (black arrows indicate hydrocarbons) and injection wells (blue arrows indicate water) drilled into oil-bearing strata deep beneath the Earth's surface. The enlargement at left shows the porous network in the oil-bearing rock. Water (blue) that has been injected under pressure into the reservoir is displacing oil (black). The regular planar grid between the surface and the reservoir represents in two dimensions the rectangular-block geometry used in bulk flow simulations of reservoir performance. Plotted on the grid are lines of constant altitude in the top layer of the reservoir, which is not parallel to the surface but rather has hills and valleys. Bulk-flow simulations of reservoir-flow performance are an important ingredient in the decision of whether to invest in major recovery projects. Improved prediction of bulk-flow parameters through understanding the physics at the pore scale is the goal of the lattice-Boltzmann simulations of multiphase flow through porous media.

In today's world oil market, economic production of oil and gas resources requires carefully engineered recovery projects of increasing technical complexity and sophistication. Hydrocarbons do not reside in cavernous pools awaiting discovery. Rather they are found—sometimes at enormous depths—within the confines of tiny pores in rock. Although the pores may be interconnected, the resulting pathways still present a significant resistance to the flow of oil toward a well drilled into the hydrocarbon-bearing strata. In addition, since water resides in some of the pores, hydrocarbons and water are recovered simultaneously at the well-head. Thus, even when large amounts of oil are known to be in a

reservoir, often only a relatively small fraction of it can be recovered with conventional pumping technology. The most common method of enhancing oil recovery, which accounts for much of the oil production in the U.S., is the injection of water at strategic locations to displace the oil toward the production wells.

Modern recovery projects of the type depicted in Figure 1 require very large capital investments. A single off-shore well drilled to a depth of 15,000 feet can cost up to \$100 million. To be successful, a project must have a sizable hydrocarbon target, the promise of extracting oil or gas at a sufficiently high rate, a strategy for water separation and disposal, and a scheme for

transportation to a refinery. As exploration efforts are driven to more remote environments, the infrastructure costs for hydrocarbon production escalate. Large potential returns are inevitably accompanied by large monetary risk.

Obviously, economic analysis must be performed before deciding whether a particular investment should be made. Computer simulations of reservoir flow performance are one of the essential ingredients in the analysis. On the basis of both field and laboratory data about the distributions of different rock types and the properties of each type, we at Mobil attempt to predict the average, or bulk, flow behavior of the fluids through the hydrocarbon-bearing rock of a reservoir. Many times our simula-

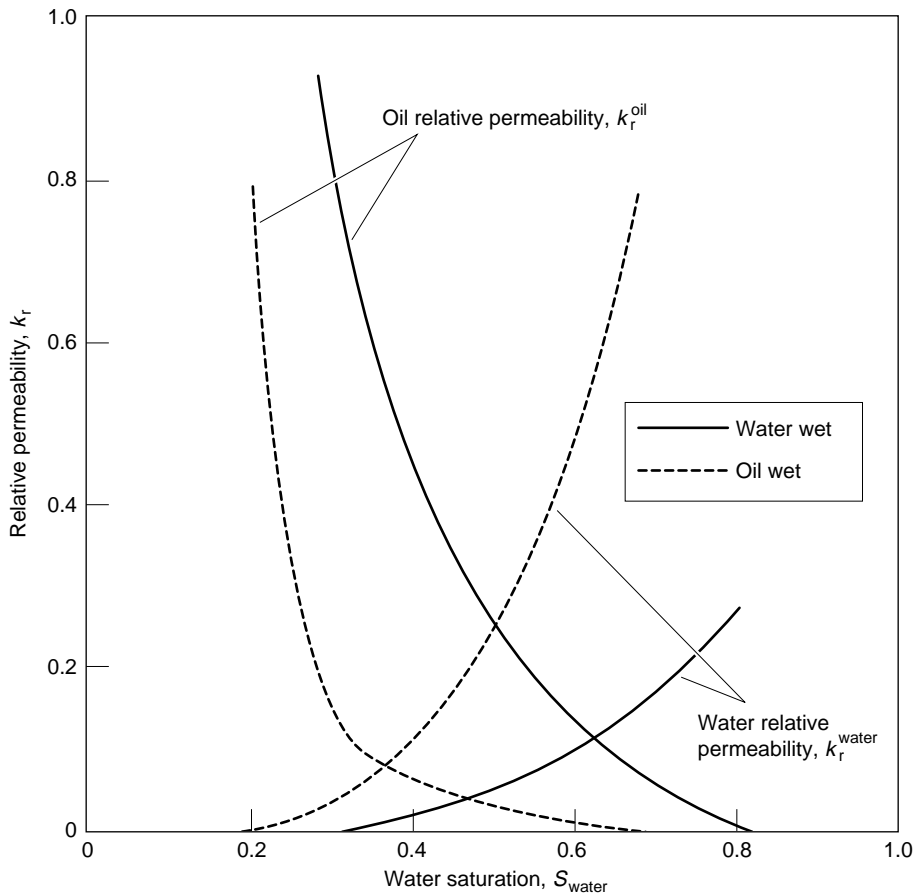


Figure 2. Typical Relative Permeability Curves for Oil and Water

Relative permeability versus water saturation (the fraction of the pore space occupied by water) is plotted for two rock types, one with a greater affinity for oil (oil-wet) and one with a greater affinity for water (water-wet). The curves were deduced from experiment. They show that oil can be recovered more easily from water-wet rock than from oil-wet rock. Note that if the rock is oil-wet (dashed lines), the relative permeability of oil falls precipitously as the fraction of the pore space occupied by oil goes below 80 percent (or equivalently, as the water saturation increases above 20 percent). For water-wet rock, the relative permeability of oil falls more gradually and not until the oil saturation goes below 70 percent.

tion-based predictions over- or underestimate reservoir performance because they do not properly model multiphase flow. Such errors can be the Achilles heel of the investment analysis, leading to very expensive errors in production strategy or even to a mistake in the decision of whether or not to invest.

The collaboration reported here between Mobil and Los Alamos National Laboratory is designed to improve reservoir-flow predictions by modeling fluid flow at the scale of individual pores, tens of microns across, and using those results to predict bulk-flow parameters. Ultimately our results will be incorporated into reservoir-scale computer simulations. Relative permeabilities are sensitive bulk parameters used in the simulations because they can change by factors of 2 or 3 depending on details of the fluid-fluid and rock-fluid interactions at the pore scale. Moreover such differences can translate into significant differences in the oil and water flow rates seen at the well-head. An improved ability to predict relative permeabilities from easy-to-gather data is one of the major goals of our collaborative effort.

Parameters for porous-media flow.

Permeabilities and relative permeabilities are defined by Darcy's law for flow through porous media. This empirical law says that the rate of flow through a porous rock is proportional to the pressure drop per unit length along the direction of flow. The constant of proportionality is called the permeability k . Specifically, the flow rate of a fluid, Q , along a direction x is proportional to the pressure gradient $\Delta p/\Delta x$ along that direction:

$$Q = \frac{kA}{\mu} \frac{\Delta p}{\Delta x},$$

where A is the cross-sectional area of

the porous rock perpendicular to the direction of flow and μ is the viscosity of the fluid. The permeability k describes the ease with which fluids flow through the porous network. Under a given pressure gradient, fluid will flow more slowly through a rock of low permeability than through one of high permeability. For the most part, the pore geometry of a rock (size and degree of connectedness of the pores) determines its permeability.

To describe the bulk flow in oil reservoirs, where pressure gradients cause oil, water, and gas to compete for space to flow, the concept of permeability is extended to include the sensitive parameters mentioned above, the relative permeabilities of each fluid present. Consider the case in which both oil and water occupy the pore space of a rock. Then the application of a pressure gradient will cause both oil and water to flow simultaneously. Darcy's law is modified to describe the flow rate of each fluid:

$$Q_{\text{total}} = Q_{\text{oil}} + Q_{\text{water}}$$

$$= (k_r^{\text{oil}} + k_r^{\text{water}}) \frac{kA}{\mu} \frac{\Delta p}{\Delta x},$$

where Q_{oil} is the flow rate of oil, Q_{water} is the flow rate of water, k_r^{oil} is the relative permeability of oil, and k_r^{water} is the relative permeability of water. Relative permeabilities depend not only on the pore geometry, which is fixed, but also on variable quantities including the saturation of each fluid (fraction of the pore space occupied by each fluid), and the spatial distribution of each fluid within the pores. The sum of the two relative permeabilities is typically less than one because fluid-fluid interactions increase the resistance to flow.

Fluid distributions at the pore scale are governed, in turn, by fluid-fluid in-

Porous medium	$k(\text{experiment})$ [darcy]	$k(\text{simulation})$ [darcy]	Imaging resolution	Comments
Micromodel-1	23	21.5	~80 μm	Pore depth = 50 μm
Micromodel-2	37	32	~80 μm	Pore depth = 74 μm
Beadpack	1100	1000	50 μm	
Berea	1	1.2	10 μm	
Sandstone	7.4	8.6	20 μm	Flow in long direction
Sandstone	7.4	21.5	20 μm	Flow in short direction

Table 1. Permeabilities Deduced from Simulation and Experiment

teractions and rock-fluid interactions. The latter determine an empirical quantity called wettability, the "preference" of a rock to be in contact with one fluid over another. In the jargon of the field, a rock is called water-wet and water is called the wetting fluid if, in the presence of water and oil, the rock surface interacts more strongly with water than with oil so that water wets (spreads out on) the rock surface and excludes direct contact with oil. A rock is called oil-wet and oil is called the wetting fluid if the rock "prefers" oil over water. At one time, experts thought all oil reservoirs were water-wet. Now most experts agree that reservoirs are predominantly of intermediate or mixed wettability. Intermediate wettability refers to the absence of any preferential interactions. Mixed wettability, on the other hand, implies that some regions are oil-wet and others water-wet on scales all the way down to individual pores.

Although it is well known that wettability influences the size and shape of pathways available for fluid flow and is therefore a controlling parameter for relative permeabilities, an explicit quantitative relationship between wettability and relative permeability has not yet been derived. Our current pore-scale studies, which include explicit modeling of fluid-fluid and fluid-rock interactions, should help to uncover that and other fundamental relationships among the

empirical parameters used to describe multiphase flow through porous media.

Current data for reservoir-scale simulations. Obtaining relative permeability data is both difficult and expensive. Nevertheless, those data are crucial to large-scale simulations of reservoir flow performance. In the simulations the reservoir is treated as a set of tens of thousands of block-shaped regions, 100 to 1000 feet on a side, with each block assigned a permeability k and relative permeability curves for oil, water, and gas. These curves typically show relative permeabilities versus saturation of one of the fluids (see Figure 2). The saturation of a fluid is the fraction of the pore space occupied by that fluid.

Relative permeabilities are derived from bench-top experiments on rock samples that are specially cut during the well-drilling process by a drill bit with a hollow core. The rock "cores" are typically 1 to 2 inches in diameter and 3 to 4 inches in length. The cores are carefully selected to represent larger sections of reservoir rock. Retrieval of such cored reservoir rock is expensive and has historically been limited to at most about a half-dozen samples per reservoir. The laboratory experiments attempt to simulate flow under subsurface conditions of the fluids found in reservoirs. Hydrocarbons and salty

water are pumped through the samples at the pressures of thousands of pounds per square inch and temperatures between 100 to 250 degrees Fahrenheit. Relative permeabilities, as defined by Darcy's law, are computed from the measured flow rates and applied pressure gradients. It is only through such experiments that the effects of wettability on multiphase flow are incorporated into the simulations. Unfortunately the laboratory experiments may not always reflect the conditions present in the subsurface. For example, the wettability of the samples may have been altered by the invasion of drilling lubricants during the sample-retrieval process. In fact, both the quality and quantity of data on permeability and relative permeability limit the accuracy of reservoir-scale simulations.

Lattice-Boltzmann simulations of pore-scale flow. The lattice-Boltzmann method was developed at the Laboratory. This method enables us to model explicitly, in the complex geometry of a porous network, the influence of fluid-fluid and fluid-rock interactions on bulk flow parameters. Details of the lattice-Boltzmann model for the flow of immiscible fluids are described in the companion article, "Lattice-Boltzmann Fluid Dynamics." The algorithm was designed to run most efficiently on large massively parallel computers such as the Laboratory's CM-5 Connection Machine. Here we present some comparisons between two- and three-dimensional lattice-Boltzmann simulations of single-fluid and two-fluid flow and corresponding laboratory experiments.

Pore geometry is a major factor determining flow in porous media, so we had to replicate realistic geometries in both the simulations and the experiments. To simulate two-dimensional flow in the laboratory, we took advantage of computerized etching techniques

to create glass micromodels of thin sections of rock. The digitized pattern used to etch each glass micromodel was also used to create the two-dimensional pore geometry for the corresponding lattice-Boltzmann computer simulation. The three-dimensional experiments involve either beadpacks (beads of various sizes packed together to approximate the geometry of porous rock) or actual core samples from reservoirs. X-ray tomography performed at the National Synchrotron Light Source at Brookhaven National Laboratory is used to capture the three-dimensional pore structure of the rock samples in the digitized form needed for the simulations. The digital rendering of the x-ray image, which is used in the simulations, can have a resolution of one micron (much less than average pore diameters, which are typically tens of microns).

Table 1 lists the results of the single-fluid simulations, which yield values for the permeability k for different types of porous media. The simulations are in good agreement with experiment except for the case of flow across the shortest dimension of the sandstone sample. The discrepancy may be due to differences in the degree of homogeneity between the simulated region and the experimental sample. One of our present research goals is to determine the size and resolution of the simulated domain required to accurately predict permeabilities and other macroscopic flow properties from the details of pore structure, material texture, and rock-fluid interactions.

In the lattice-Boltzmann model for the flow of immiscible fluids, the viscosity of each fluid may be varied independently. Also, the surface tension at fluid-fluid interfaces and the contact angle of each fluid with the rock surface may be varied. (See Figure 6 in "Lattice-Boltzmann Fluid Dynamics")

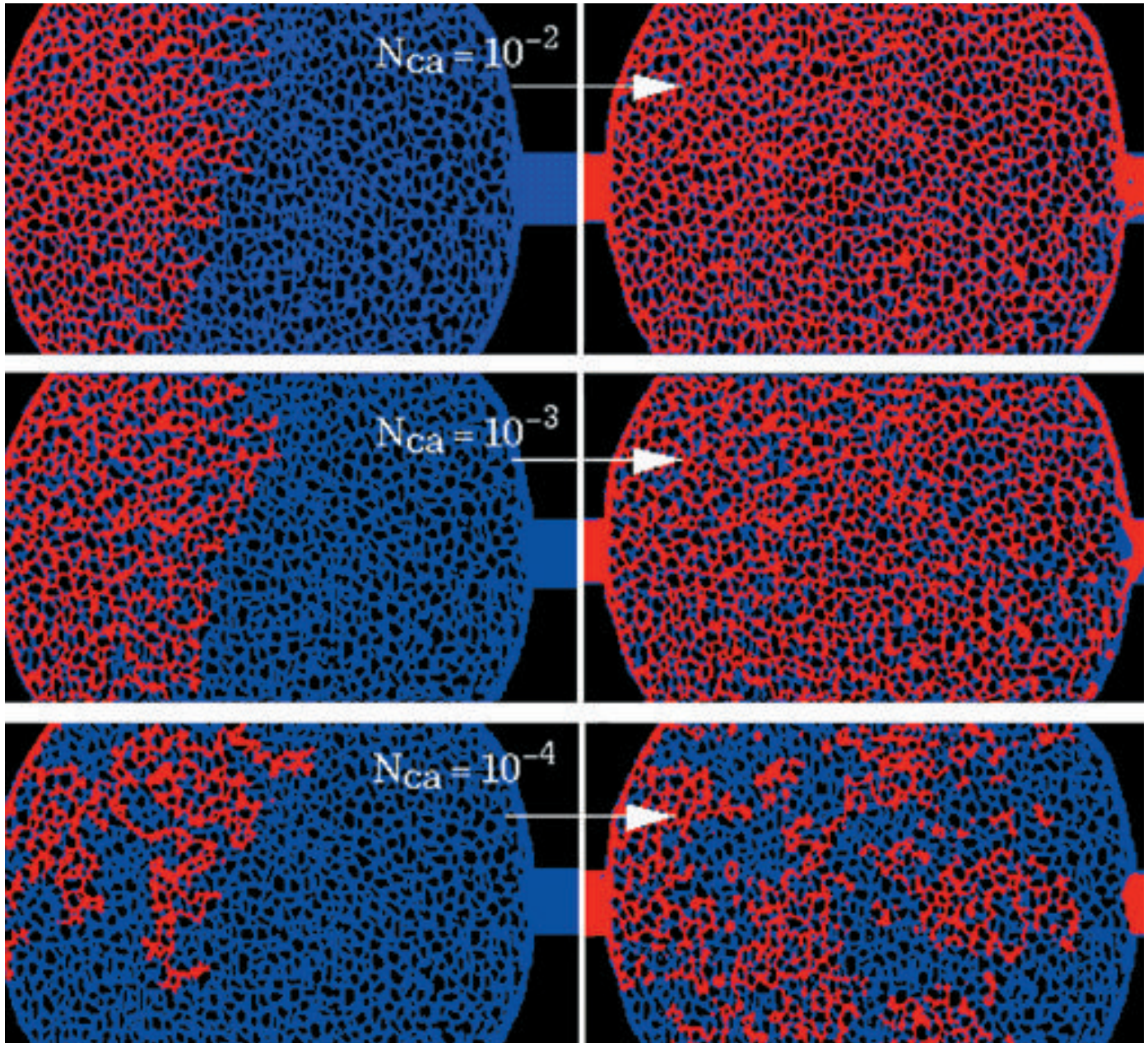


Figure 3. Water Displacing Oil at Different Capillary Numbers

These two-dimensional lattice-Boltzmann simulations show flow through a two-dimensional glass micromodel for three values of the capillary number, the ratio of viscous to interfacial (or capillary) forces. Here capillary number is defined as $N_{ca} = \mu U \phi \sigma$, where μ is viscosity, U is displacement velocity, and ϕ and σ are porosity and interfacial tension, respectively. The pores are originally filled with the blue wetting fluid (oil). The red non-wetting fluid (water) is injected at left. Different displacement velocities were used to simulate the effect of different capillary numbers on the efficiency with which the pores are swept. Note that as the velocity decreases, or the capillary number decreases, the red non-wetting fluid (water) becomes less effective at displacing the blue wetting fluid (oil). At low capillary number, when interfacial forces dominate, the simulation produces an irregular displacement front. These results are realistic compared with published laboratory results.

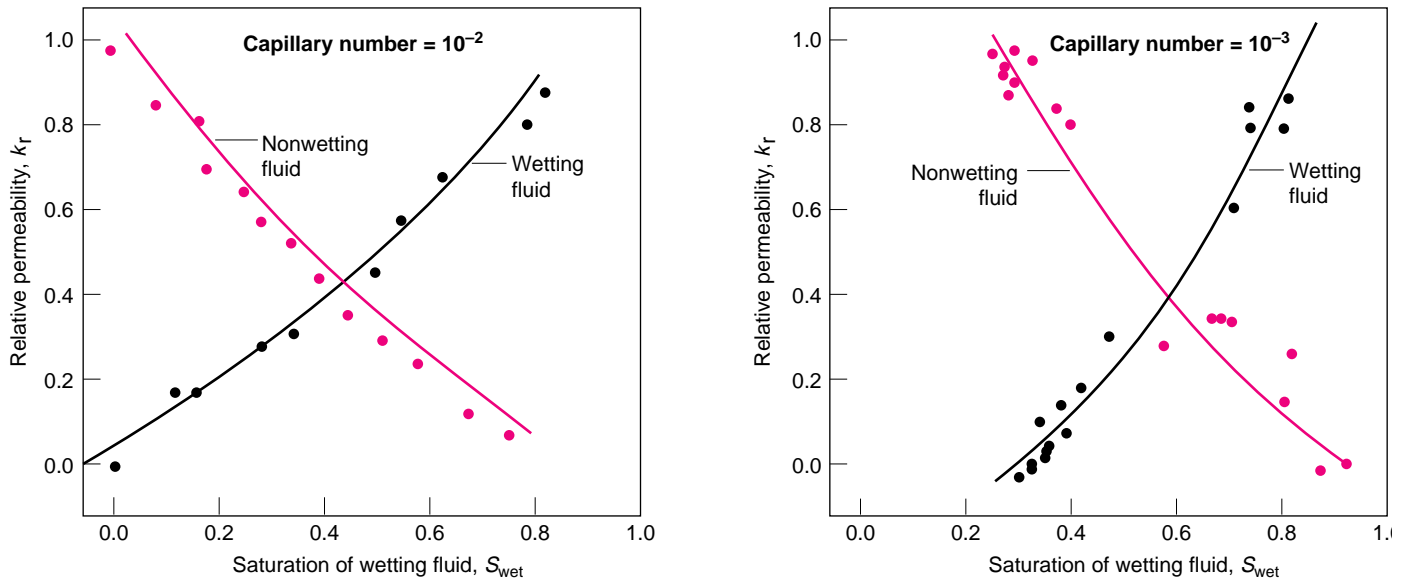


Figure 4. Relative Permeability Curves for Oil and Water Predicted by Lattice-Boltzmann Simulations

These two plots show relative permeabilities k_r for the wetting fluid (oil) and the nonwetting fluid (water) as a function of the wetting-fluid saturation for two values of the capillary number. Each data point was derived from lattice-Boltzmann simulations of flow through the network shown in Figure 3. The curves are similar to those obtained in the laboratory experiments. The relative permeability of the wetting fluid increases, that is, the fluid flows more easily, as its saturation (the fraction of pore space it occupies) increases. Similarly, the relative permeability of the nonwetting fluid decreases as its saturation (equal to $1 - S_{wet}$) decreases. Note how different values of the capillary number change the shape and position of the curves.

for the definition of contact angle.) The contact angle of a fluid is 0° if the fluid completely wets the rock and increases to 180° as the fluid changes from strongly wetting to strongly nonwetting. In lattice-Boltzmann simulations we vary the strength of rock-fluid interactions to control the contact angle and thus the wettability conditions.

One quantity used to characterize the flow is capillary number—the ratio of the viscous forces (the shear forces within each fluid) to the forces at the fluid-fluid interfaces. When the capillary number is low, interfacial forces are much stronger than viscous forces and tend to control the flow. For example, the interfacial forces will tend to draw the wetting fluid into the smallest pores and thereby block the invasion of the nonwetting fluid. We are using

lattice-Boltzmann simulations to determine the quantitative effects of wettability and capillary number on relative permeabilities and displacement efficiency (the efficiency of displacing one fluid by injection of a second fluid).

Figure 3 shows the results of a two-dimensional simulation for three values of the capillary number. The pores are initially filled with the blue wetting fluid (oil) and are being invaded by the red nonwetting fluid (water). The simulation reproduces the correct qualitative relation between capillary number and oil recovery efficiency under oil-wet conditions—namely, the recovery becomes less efficient as the capillary number decreases. In part (a), where the capillary number is highest, viscous forces dominate and we see more uniform or piston-like displacement of the

oil and higher displacement efficiency. In part (c), viscous forces are a hundred times smaller, so capillary forces dominate and the displacement front has many fingers. Also, the displacement efficiency is much lower than in (a). The blue (oil) and red (water) distributions in Figure 3 are similar to the oil and water distributions observed by nuclear-magnetic-resonance micro-imaging of flow through a mixture of glass and polystyrene beads.

Relative permeabilities of the pore network in Figure 3 under various conditions can be deduced from the results of lattice-Boltzmann simulations. In particular, Figure 4 shows the relative permeabilities of the wetting (blue/oil) and nonwetting (red/water) fluids versus the saturation of the wetting fluid for two of the three values of the capillary

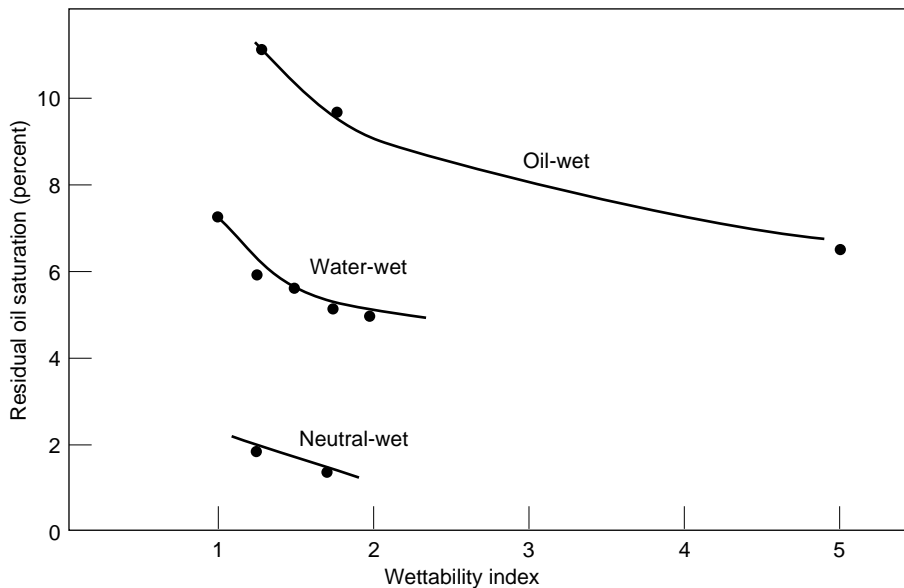


Figure 5. Residual Oil Saturation and Wettability

These data were obtained from lattice-gas simulations run on the system shown in Figure 2. The residual oil saturation is the percentage of pore space occupied by oil after the oil-filled pores have been flushed with water. The wettability conditions are varied by changing the collision rules at the fluid-rock boundaries. The rules implement slip or non-slip boundary conditions for all particles near a rock surface. Intermediate wettabilities are simulated by random fluctuations between the two types of boundary conditions. The wettability index is a dimensionless quantity that measures the relative affinity of a rock sample to draw in oil or water spontaneously.

number used in our simulations. Again, the curves are typical of those seen in the laboratory. The input to our simulations includes pore geometry, fluid viscosities, and interfacial tensions. The capability to predict relative permeability, an empirical quantity, from such a simulation is new and is still being validated against experimental flow data.

Although we are still validating the lattice-Boltzmann simulation technique, we have already begun a study of the effects of wettability on two-fluid flow in two- and three-dimensional simulations. Figure 5 shows the residual oil saturation (the percentage of the oil that remains after being flushed with water) versus wettability for oil-wet, water-

wet, and neutral-wet conditions as obtained from some early two-dimensional simulations using lattice-gas rather than lattice-Boltzmann methods. As expected, the simulations show that the most efficient displacement of oil by water is obtained at neutral wettability, whereas it is most difficult to extract oil if the rock is oil-wet. A major input to such simulations are the rock-fluid interactions that define the wettability conditions. At Mobil we are developing techniques to map wettability on the basis of mineral distribution and fluid properties, so that we can incorporate realistic complex wettability distributions into our simulations. Figure 6 shows a typical three-dimensional lattice-Boltzmann simulation in which

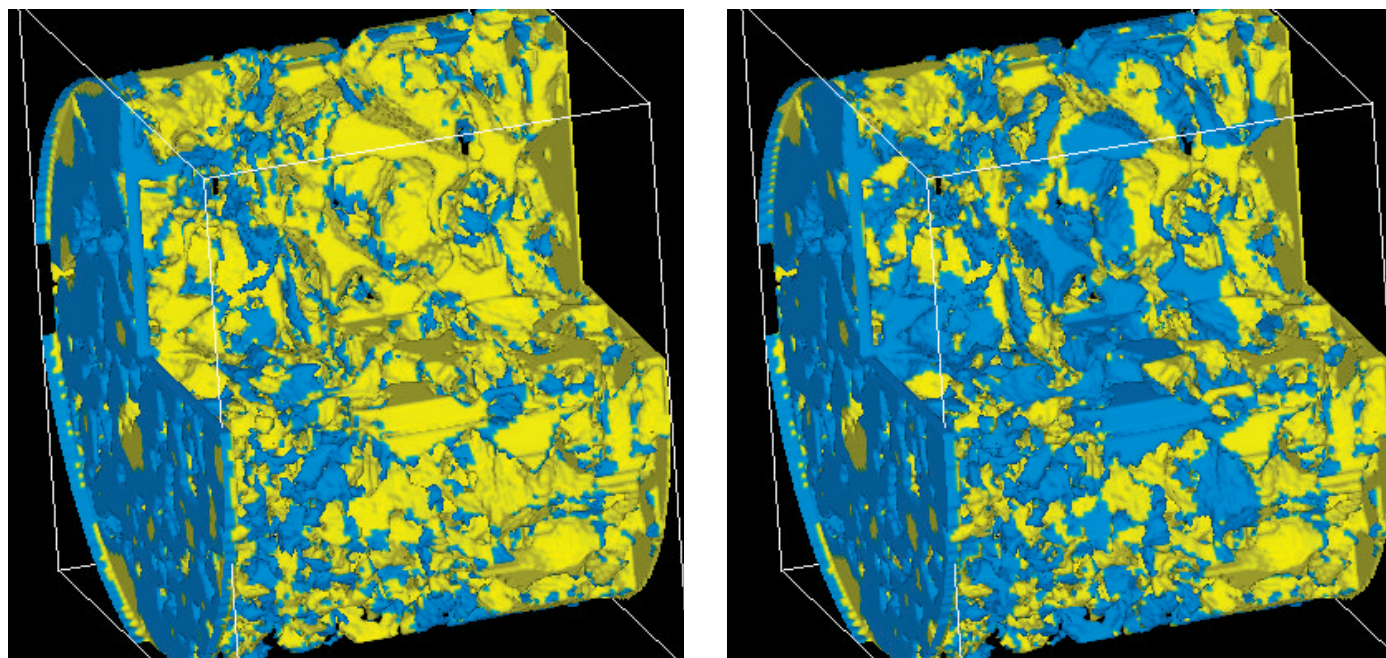
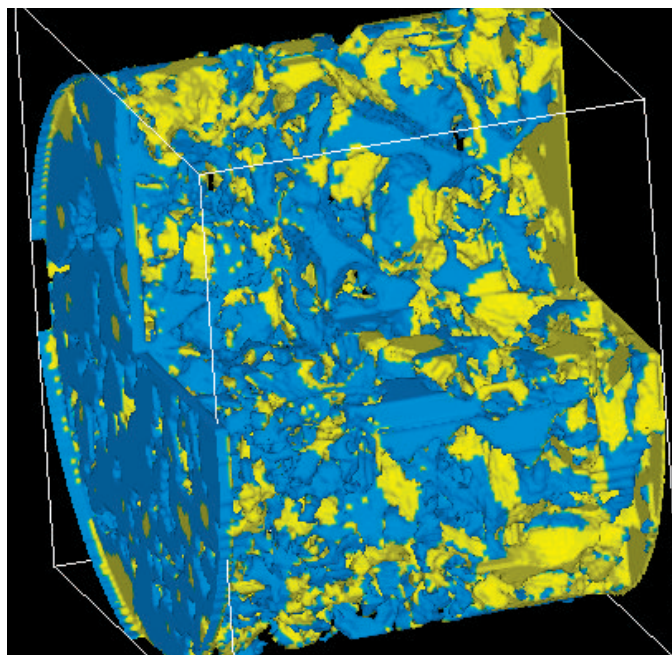


Figure 6. Water Displacing Oil in Three Dimensions
This figure shows snapshots taken at different timesteps during a three-dimensional lattice-Boltzmann simulation of the forced displacement of oil (yellow) by water (blue) within a porous rock. The pore geometry in this simulation replicates that in a portion of a rock sample obtained from a Mobil offshore oil reservoir over 10,000 feet below sea level. The geometry was determined from a computed tomography (CT) scan that provided a three-dimensional digital image of the pore structure at a resolution of 20 microns. The rock is shown as transparent in order to make visible the oil and water occupying the pores. The second and third snapshots show that water continually displaces oil as the simulation progresses.



water is displacing oil in a water-wet system. During this simulated displacement, we recover bulk flow properties, such as mass fluxes, local velocities, and pressure profiles, from which we can compute relative permeabilities. Such computer experiments provide far more insight into and control over the displacement process than wet-lab experiments.

The state-of-the-art technology resources required for such a project are beyond what private companies—even large ones—can afford. The Laboratory and Mobil Corporation have combined their expertise and their resources to apply Laboratory-developed lattice-Boltzmann technology to tough research problems in the oil industry. We expect the results of this work to have major implications as U.S. reservoirs mature and require more complex and risky recovery schemes. ■

Further Reading

I. Field Roebuck, Jr. 1979. *Applied Petroleum Reservoir Technology*. IED Exploration, Inc.

H. C. Slider. 1983. *Worldwide Practical Petroleum Reservoir Engineering Methods*. PennWell Publishing Company.

L. F. Koederitz, A. H. Harvey, and M. Honarpour. 1989. *Introduction to Petroleum Reservoir Analysis*. Contributions in Petroleum Geology and Engineering, volume 6. Gulf Publishing Company.

F. A. L. Dullien. 1992. *Porous Media: Fluid Transport and Pore Structure*, second edition. Academic Press.



Standing (left to right): John J. Buckles, Shiyi Chen, and Daryl W. Grunau; seated: Randy D. Hazlett and Mary E. Coles

John J. Buckles earned his B.S. in chemical engineering from Purdue University in 1976 and his Ph.D. in chemical engineering from the University of Illinois in 1983. For the past eleven years he has been employed at Mobil Research and Development Corporation. Buckles's research interests include enhanced oil recovery and pore-scale flow modeling.

Daryl W. Grunau was a graduate research assistant at the Laboratory in 1990 and again from 1991 to 1993. His work included the development of theory and computer code for research in computational fluid dynamics using the lattice-gas and lattice-Boltzmann methods. He is currently an applications engineer at Thinking Machines Corporation. Grunau received his B.A. in mathematics and computer science from Tabor College in Kansas in 1987 and his M.S. and Ph.D. in applied mathematics from Colorado State University in 1989 and 1993.

Randy D. Hazlett earned his B.S. and Ph.D. in chemical engineering from the University of Texas, Austin, in 1980 and 1986. Since 1986 he has been employed at Mobil Research and Development Corporation. Hazlett's research interests include flow in porous media, interfacial phenomena, improved oil recovery, and imaging and image processing.

Wendy E. Soll has been a staff member of the Geoanalysis Group in the Earth and Environmental Sciences Division since 1993. She was a postdoctoral fellow at the Laboratory from 1992 to 1993. Her research includes the study of multiphase flow and transport in porous media utilizing lattice-Boltzmann pore-scale modeling. She received her B.S.E. in mechanical engineering from Princeton University in 1984. At the Massachusetts Institute of Technology she earned her M.S. in mechanical engineering in 1987 and her Sc.D. in civil engineering in 1991. Soll is a member of the American Geophysical Union, the Society of Petroleum Engineers, and the American Society of Mechanical Engineers.

The biographies of co-authors Shiyi Chen and Kenneth G. Eggert appear on page 109.

CONCEPT EXTRACTION

a data-mining technique

*Vance Faber, Judith G. Hochberg, Patrick M. Kelly,
Timothy R. Thomas, and James M. White*

Just as miners must process huge quantities of rock and dirt to obtain valuable ores, data analysts must often process huge volumes of raw data to extract useful information.

The use of computers in numerous applications is generating data at a rate that far outstrips our ability to process and analyze it. For example, NASA satellites are expected to generate hundreds of terabytes of data per day (1 terabyte = 10^{12} bytes). Sets of financial data ranging from credit-card transactions to shipping records contain terabytes, and textual databases are growing rapidly. A great deal of effort is currently being expended to develop new hardware and software to generate, transmit, and store such data, but relatively little emphasis has been placed on developing new ways to use computers to analyze the data after they are acquired.

“Data mining,” or the process of extracting useful information from very large datasets, is a focus of our efforts in the Laboratory’s Computer Research and Applications Group. In this article we describe a data-mining method we call concept extraction, which involves both humans and computers in a productive partnership. Here “concept” is defined as a psycholinguistic category that can be either physical (*animal* or *blue*) or abstract (*mood* or *politics*).

The underlying premise of concept extraction is that in order to interpret data, humans naturally and quickly extract and identify significant concepts. A person can contemplate a landscape, receive data through the sense organs, and can then make a reasonable judgment about whether it will rain, for example, or whether it will snow. A person can scan a magazine’s table of contents and easily select the articles that relate to a subject of interest. Humans are constantly assimilating, categorizing, synthesizing, and analyzing data—most often without any conscious realization of doing so.

The ability to extract concepts from sensory data is the product of millions of years of evolution and years of indi-

vidual learning and experience. But humans go beyond simply recognizing and naming objects or events; we make judgements based on an overall context or quite subtle correlations among diverse elements of the available data. The great complexity, subjectivity, and ambiguity of human concepts make them extremely difficult if not impossible to define in a quantitative manner appropriate for use by computers. As linguist William Labov puts it, “Words that are bound to simple conjunctive definitions will have little value for application in a world which presents us with an unlimited range of new and variable objects for description.”

Computers can, however, make data mining easier because they can quickly and accurately perform certain tasks that demand of humans too much time or concentration. For example, an expert in the interpretation of satellite images may wish to determine the ratio of urban to rural land use in a particular region. No matter how skilled the expert may be at distinguishing between the urban and rural areas in the image, the task of identifying each area in a large image is time-consuming and tedious. In contrast, computers—once appropriately programmed—are ideally suited to that task. We have accelerated the concept-extraction process by breaking it down into quantitative and intuitive portions, and assigning the former to computers and the latter to human experts.

First, the computer reduces the size of the dataset composing the satellite image while retaining its essential character. This crucial step employs a new, high-speed clustering algorithm specifically developed to handle very large datasets. The human then identifies, or extracts, an example of each concept of interest, in this case an urban area and a rural area, within the image produced from the reduced dataset. An expert

ability such as this cannot readily be translated into a computer program, but after examples have been provided, the computer can identify any repetitions of the examples in the image.

Our method greatly increases the rate at which an expert, or even a novice, can analyze a large and complex dataset. Concept extraction is not an exact process—even an expert can be inconsistent or make mistakes. But by enabling the computer to approximate the expert’s interpretive skills, concept extraction provides a flexible, rapid way to incorporate the human perspective—flaws and all—into computer analysis.

Concept Extraction Applied to Satellite Images

Traditional analysis methods. Images produced by satellites orbiting Earth serve a variety of purposes including assessing weather conditions, such as heavy rains likely to cause flooding, and tracking long-term environmental trends, such as deforestation. Satellite images are, of course, also a source of military intelligence. Here we focus on the application of concept extraction to data obtained by the Landsat 4 system, an unclassified system that has been in operation since 1982.

Traditional image analysis requires an expert for two reasons. First, the analyst must interpret Earth data from a rather unusual perspective—imagine the difference between an eye-level view of a forest and a satellite view from an altitude of 450 miles. Second, the analyst must understand the significance of all the recorded information, which consists of the intensities of the radiation reflected from or emitted by the surface of the earth. Intensity measurements are recorded for seven regions of the electromagnetic spectrum, three

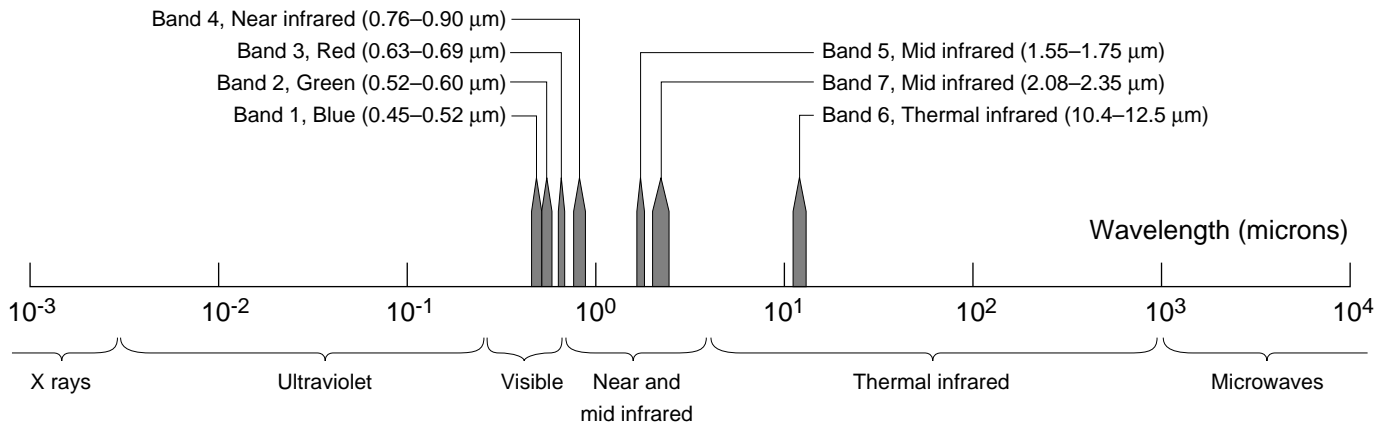


Figure 1. Locations within the Electromagnetic Spectrum of the Seven Bands Recorded by the Landsat System A portion of the electromagnetic spectrum, the continuum of all electromagnetic waves, is arranged from left to right according to increasing wavelengths. Landsat sensors collect data in seven spectral bands: three in the visible portion of the spectrum (Bands 1, 2, and 3); one in the near-infrared (Band 4); two in the mid-infrared (Bands 5 and 7); and one in the thermal infrared (Band 6).

in the visible region (red, blue, and green) and four in the infrared region (see Figure 1). Analysts must know which spectral regions help identify which surface features and must locate and identify each separate occurrence of each concept. Even an expert analyst may need up to several weeks to extract the desired information from a single Landsat 4 image.

The slow pace of the analysis is mainly due to the enormous size of the dataset. A typical Landsat image covers an area of about 10,000 square miles, or 100 miles on a side. The digital image is composed of about 50 million pixels, or 7000 pixels per side, so each pixel represents a square region of about 75 feet on a side. For each pixel the Landsat system measures the radiation intensity in each of the seven spectral bands and stores each of the measured intensities as an 8-bit (or 1-byte) number. So, the data for a single image amount to 7 1-byte numbers per pixel, or a total of roughly 350 million bytes of data.

These data are typically viewed on a high-cost, 24-bit-per-pixel color screen. To produce an ordinary color image, the red, green, and blue 8-bit intensity values for each pixel of the recorded image are mapped to the blue, green,

and red components that constitute each pixel on the screen (see Figure 2). To gain a different perspective and thus additional information, analysts often display other combinations of spectral bands (see Figure 3).

Even a simple mapping of the spectral data to the display screen requires time-consuming computation. The color value for each pixel must be computed separately and recomputed each time a new set of three bands is chosen for display. Usually only a quarter of an image is processed at a time, but processing even 12 million pixels takes long enough—up to a few minutes—that interactive use of the data is impractical. For more complex mappings, where two or more spectral bands might be combined according to some useful function, computations can take up to a few hours.

Data clustering—the first step in concept extraction. One of our goals in developing the concept-extraction technique was to facilitate interactive analysis of Landsat data by reducing the time required for analysis from days to hours.

The essential first step in achieving that goal was to reduce the size of the

original dataset by applying a newly developed, high-speed computer algorithm for clustering the data. (See “Clustering and the Continuous k -Means Algorithm.”) Clustering allows us to replace the original spectral data with an appropriate set of representative values. To find those values, the seven intensity values for each of the 50 million pixels are regarded as the components of a vector in a seven-dimensional spectral-intensity space. That is, each axis in the space corresponds to one of the seven spectral bands, and the coordinates specifying the end point of each vector are the seven spectral intensity values of the pixel that vector represents. The k -means algorithm groups the 50 million points into k clusters such that all the points in each cluster are more similar (“closer”) to one another than to those in the other clusters. For the Landsat data we chose a k value of 256 because 256 is the greatest number of distinct colors that can be represented on an inexpensive, 8-bit screen; we often use larger k values for other applications. The algorithm also determines each cluster’s centroid, which is the cluster’s “mean point,” or the point each of whose coordinates is the arithmetic average of the corre-

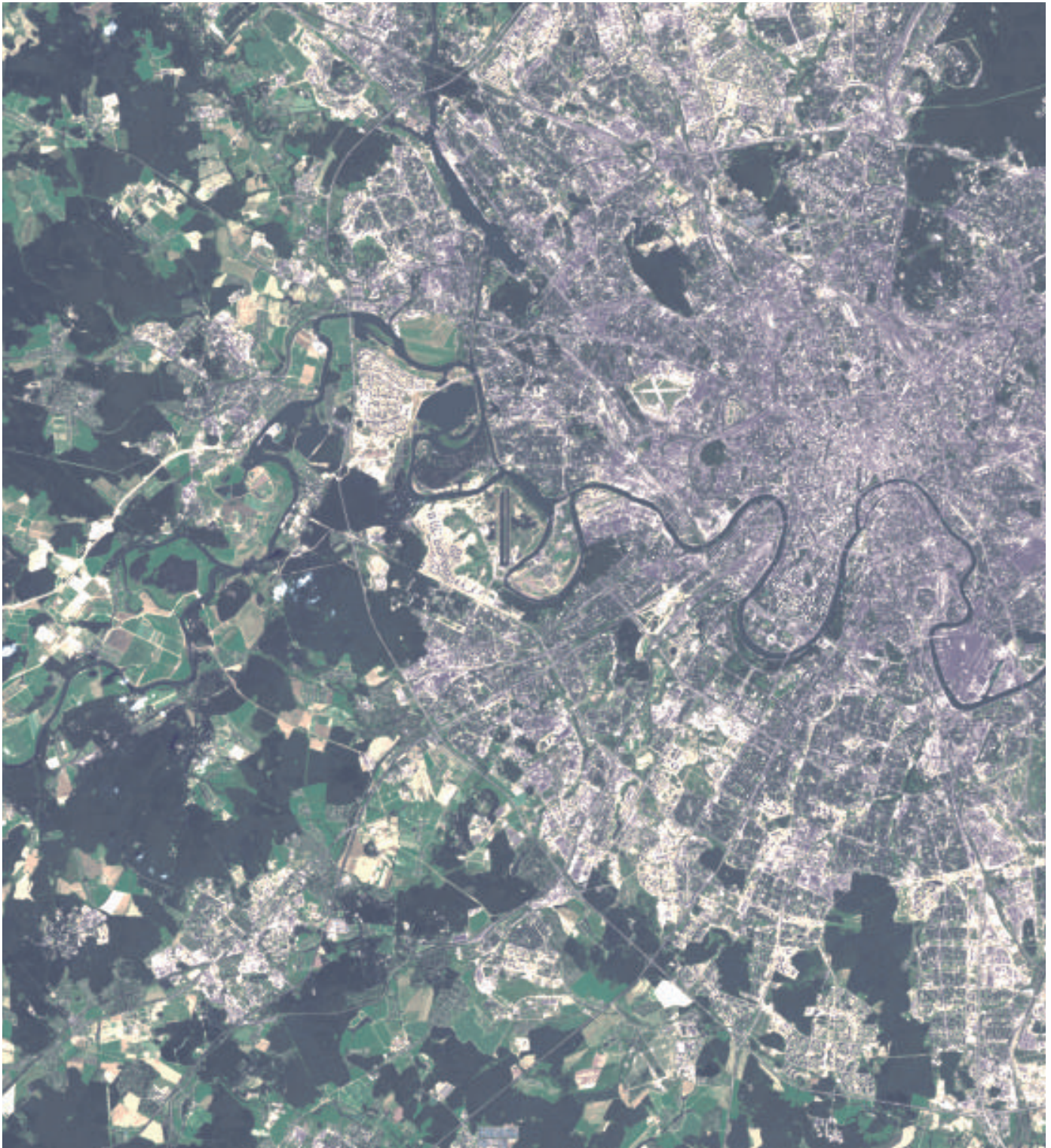


Figure 2. Landsat Image of Moscow and Its Environs

This image of Moscow and environs is based on data obtained by a Landsat satellite late in the growing season. The satellite, in orbit at an altitude of 450 miles, measured the intensities, in seven spectral bands, of the radiation reflected from or emitted by the surface of the earth. This is a small portion of the entire 50-million-pixel image, only about 800,000 pixels, and each pixel represents a square region of about 75 feet on a side. The image is an ordinary, or “natural-color,” image, and the color of each pixel is determined by combinations of intensity values in the blue, green, and red spectral bands mapped to the blue, green, and red components of each pixel. However, the intensity values combined to produce the image shown are not those measured by the satellite but, rather, are those resulting from a data-reduction technique called clustering, as discussed in the main text. The image is reproduced as it would appear on an 8-bit-per-pixel color screen.

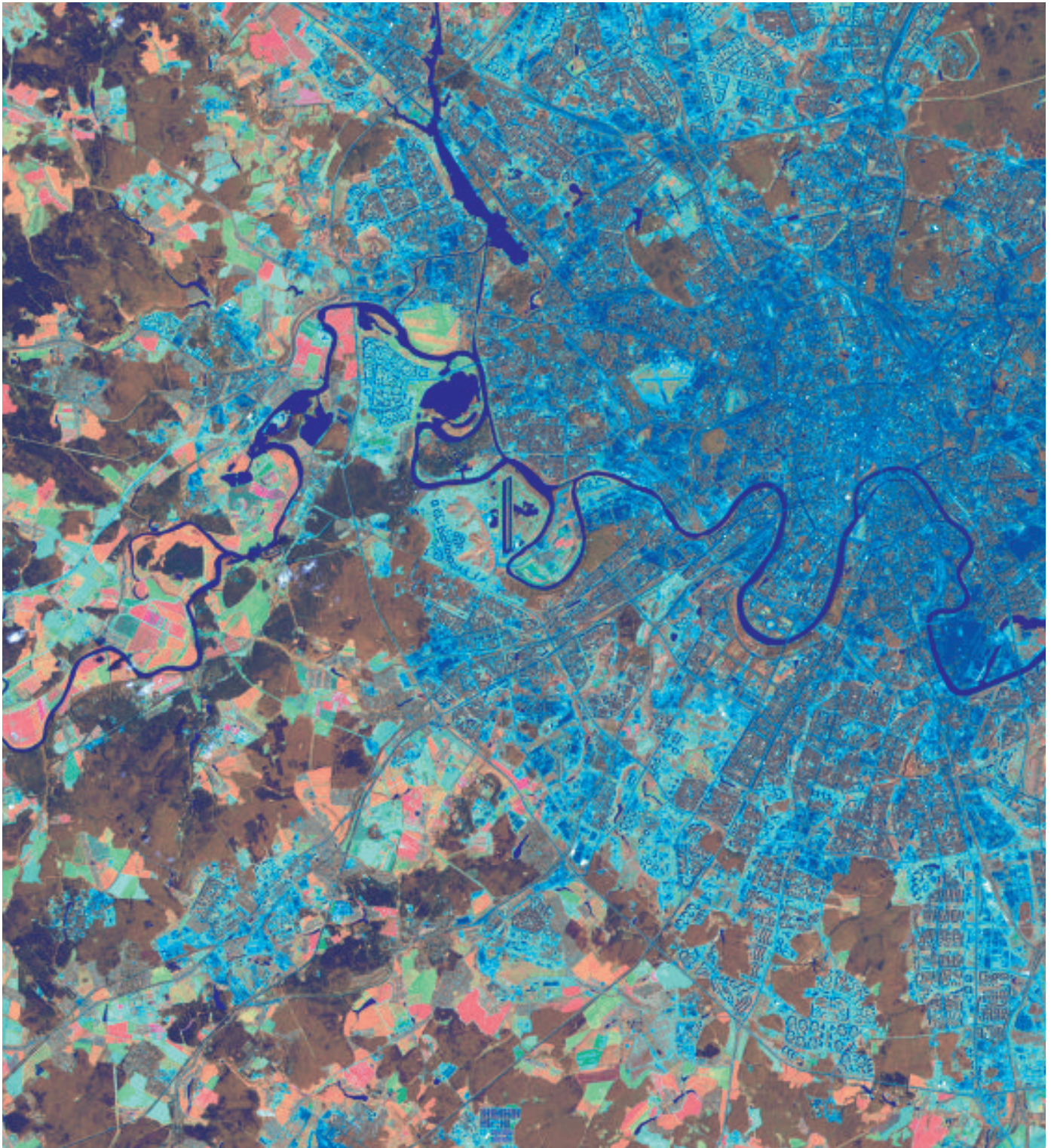


Figure 3. An Alternative Imaging of Moscow and Its Environs

This image was produced from the same reduced dataset that yielded the image in Figure 2. Here, however, is an alternative representation that yields optimal visual separation of the various types of vegetation and, in particular, agricultural versus non-agricultural regions. The color of each pixel was determined by a combination of intensities from the near-infrared Band 4, mid-infrared Band 5, and the visible Band 3 (red) mapped to the red, green, and blue pixel components respectively. With this band combination, vegetation types are differentiated by variations in both color and color intensity. The regions of various shades of red and orange are fields of growing crops. Regions with the healthiest vegetation, or greatest amount of “biomass,” appear in the most intense shades of red. The pale blue regions are unplanted fields. The brown regions represent forested areas; deciduous forests are light brown, and coniferous forests are a darker shade of brown.

Clustering Spectral Intensity Data

sponding coordinates of all the points in that cluster. Figure 4 illustrates the process of clustering spectral-intensity data for a simplified case that considers only two bands, red and blue, and thus a spectral-intensity space of only two dimensions.

After the Landsat data is grouped by the algorithm into 256 clusters, each cluster is designated by a 1-byte number from 0 to 255 and those designations are stored in a codebook along with the seven intensity values for the centroid of each cluster. Each pixel in the image is now associated with, or belongs to, a spectral cluster—the cluster into which its spectral data have been grouped. Furthermore, each pixel will now be “colored” according to the intensity values of that cluster’s centroid rather than according to the original spectral data. Figure 5 illustrates the stored data array before and after clustering.

The centroid data stored in the codebook are used to approximate the original spectral data. To produce a color image, any combination of up to three of the seven spectral bands—or any mathematical transformation of all seven—are mapped to the three components of each pixel on a color screen. Although clustering drastically reduces the amount of spectral data to the 256 centroid values, that number is substantially greater than the number of concepts (urban, rural, forest, desert, and so on) we intend to extract from the data. Thus the data still have enough detail to allow clear distinctions among those concepts. In fact, the human eye cannot distinguish a display of the clustered data from a display of the original data.

Storage and handling of clustered data. The use of clustered data greatly enhances the efficiency of all future data handling. Since the amount of

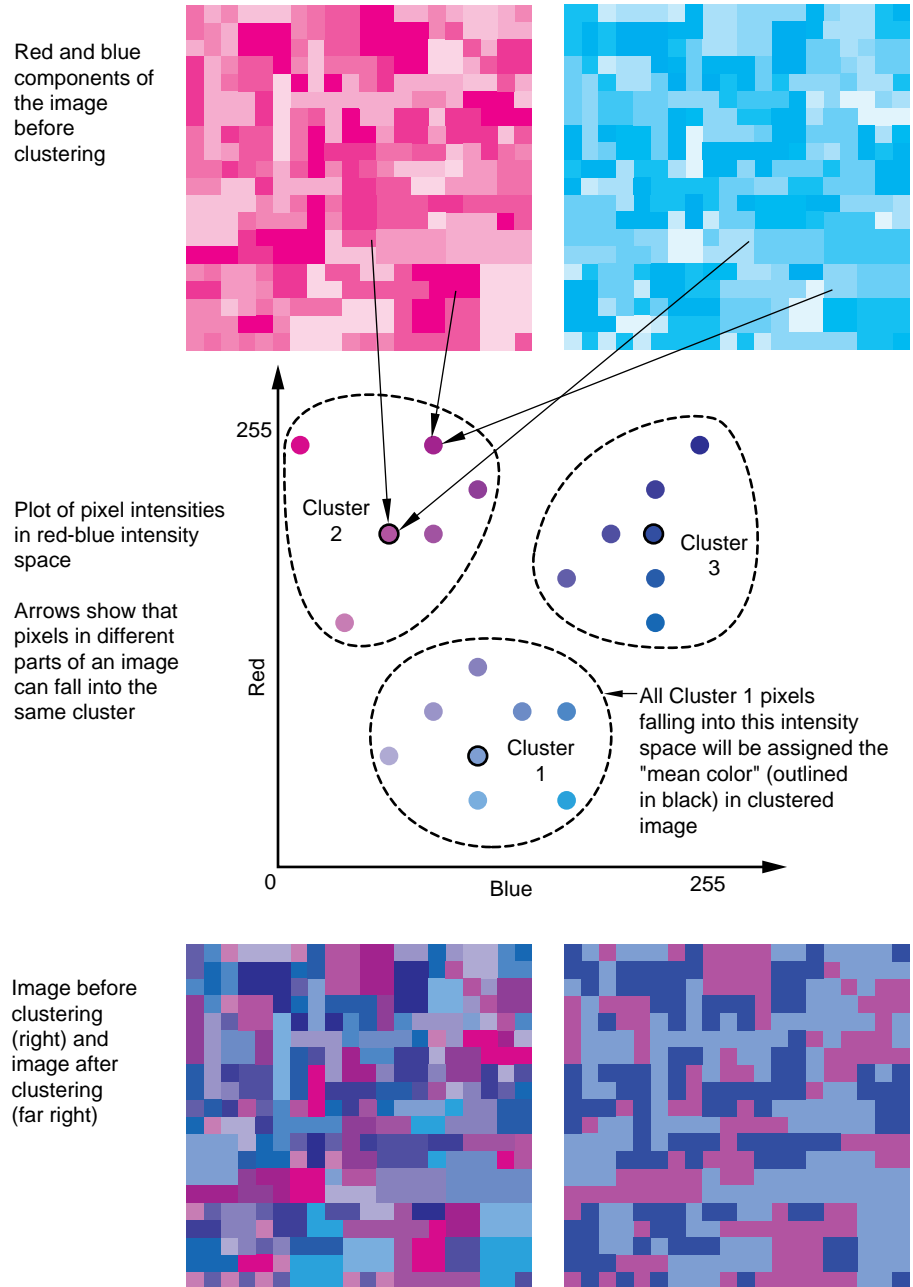
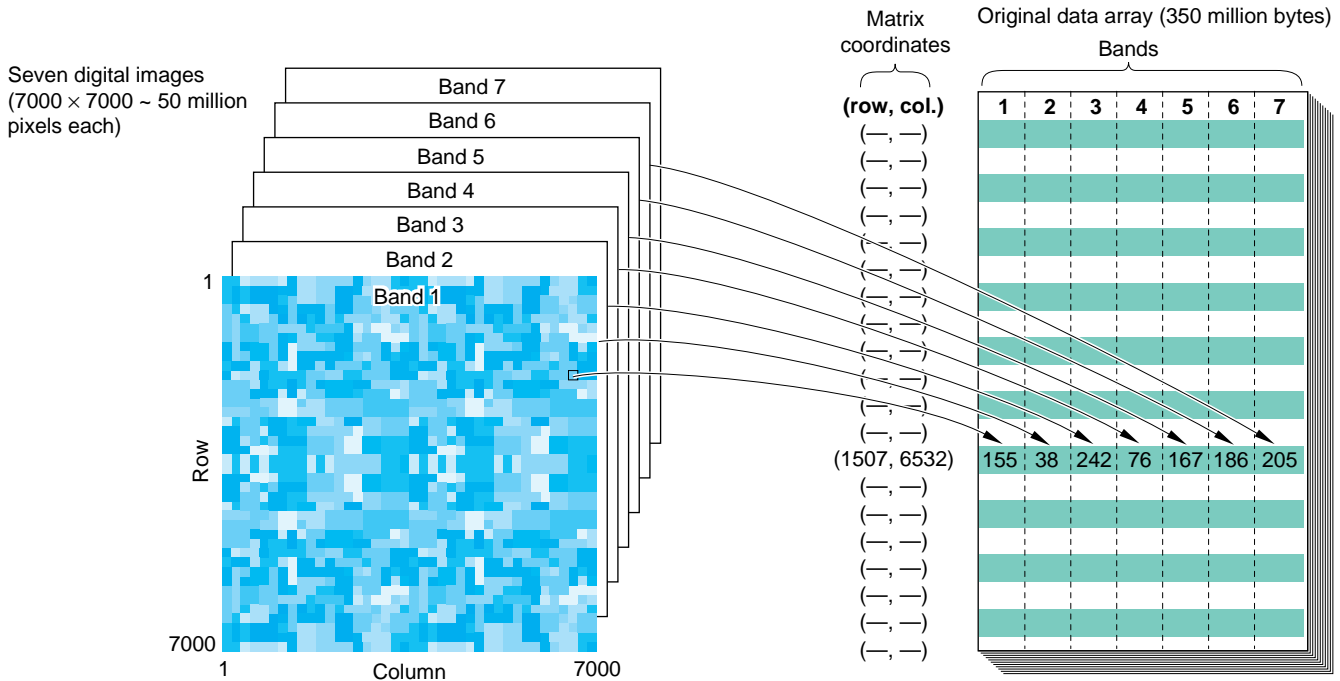


Figure 4. Clustering Landsat Data

The figure illustrates the clustering of spectral-intensity data for a simple two-band image. The red and blue components of the digital image are shown separately. Each pixel is plotted in “spectral-intensity space”; that is, the coordinates of each pixel in that space are the red and blue intensities of that pixel in the original image. The data in spectral-intensity space have been grouped into three clusters. Each of the three points outlined in black represents the centroid of its cluster, that is, the average of the coordinates of the data points in each cluster. Thus the color given by the centroid’s coordinates is the “mean color” of the cluster. When the image is displayed after the spectral data have been clustered, each pixel that belongs to cluster 1 in spectral-intensity space is colored with the centroid color of cluster 1, and similarly the pixels that belong to other clusters are colored with the centroid color of their clusters. Thus in this simplified example the final image is composed of pixels with just three colors. In reality, the data for Landsat images are grouped into 256 clusters, and the resulting 256-color images cannot be visually distinguished from the original Landsat images.

(a) Original data and data array before clustering



(b) Data array and codebook after clustering

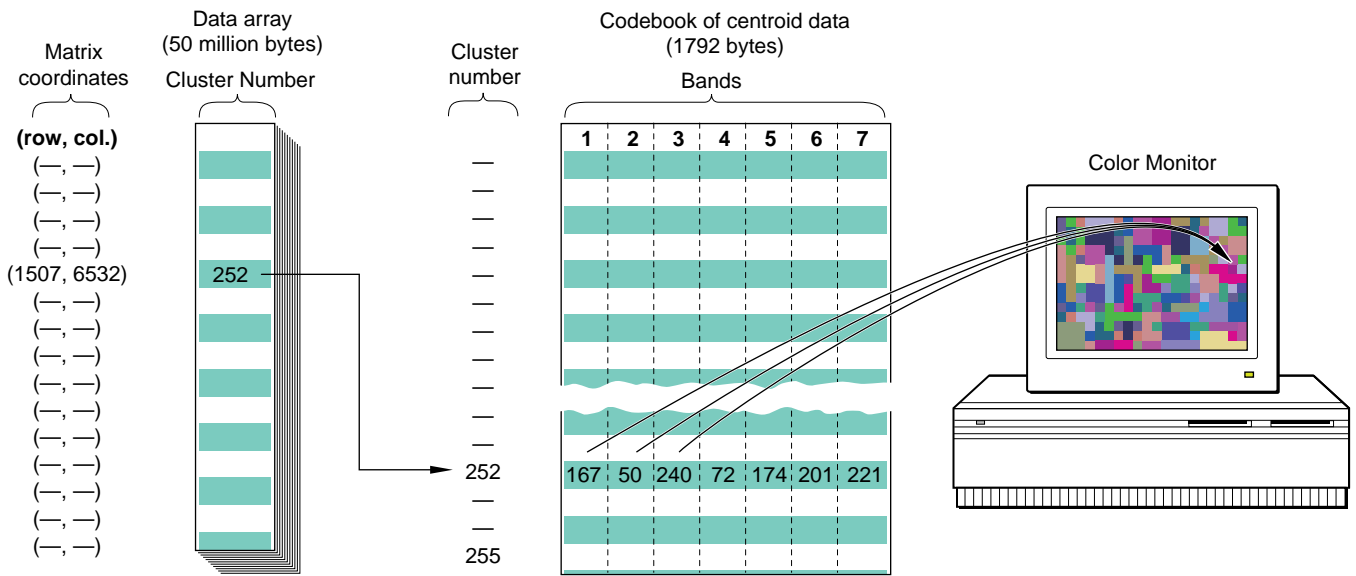
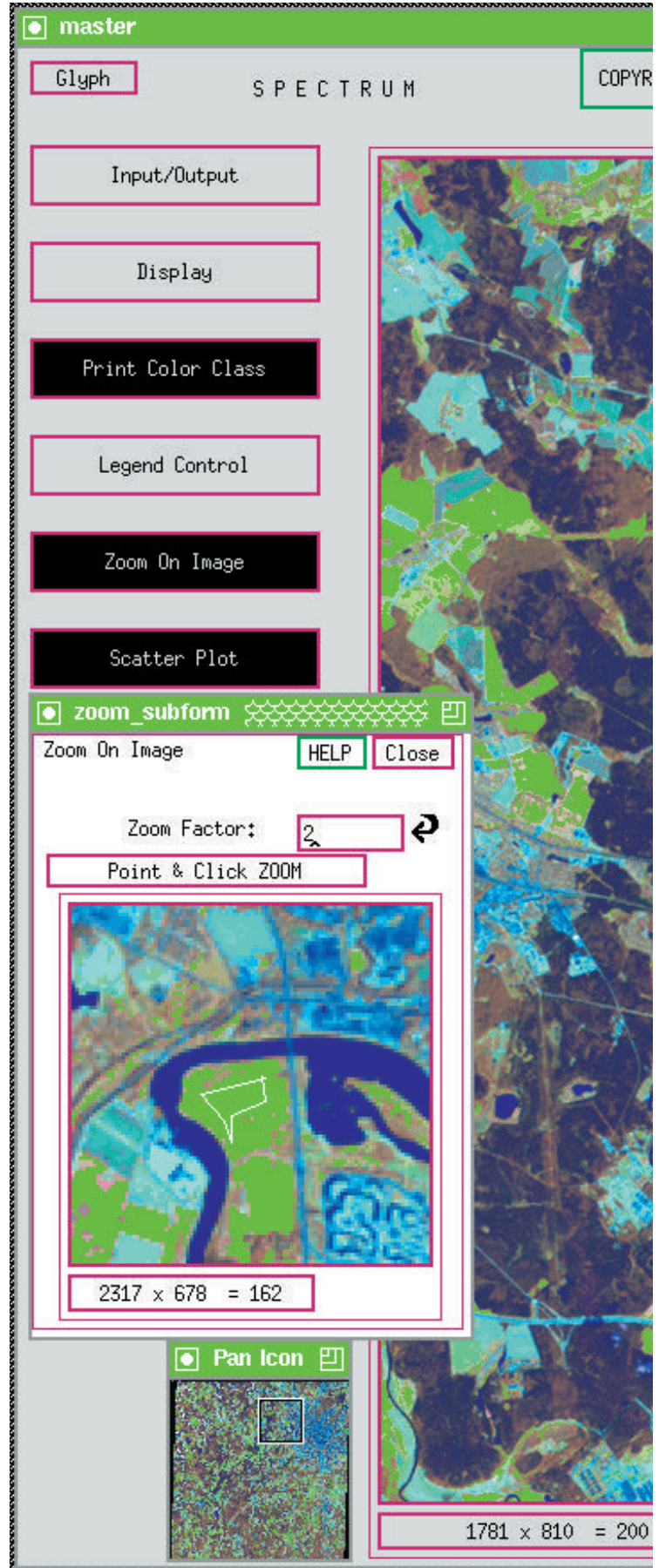


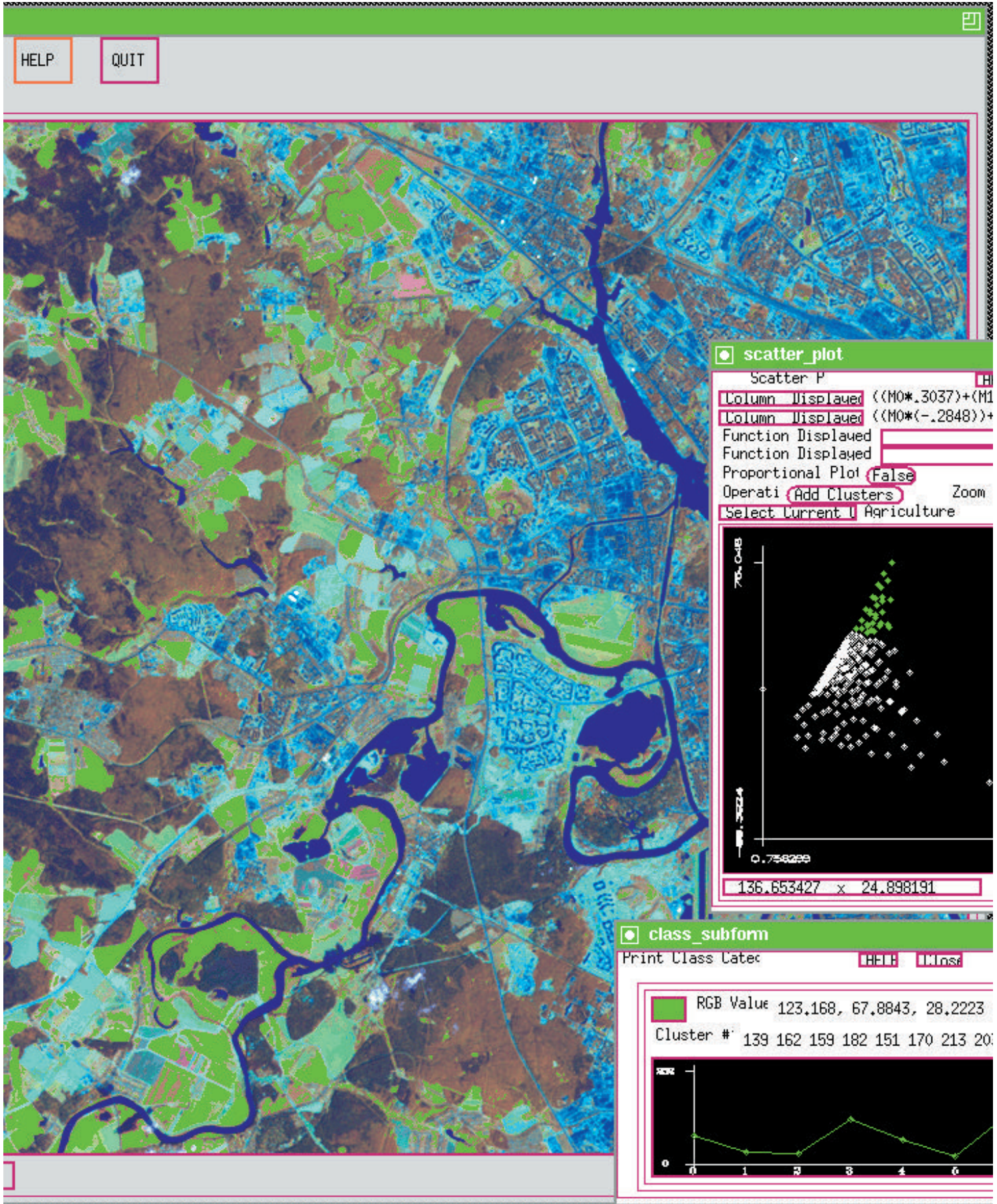
Figure 5. The Data Array before and after Clustering

(a) The original data for a Landsat image are represented as a set of seven digital images, one for each of the seven spectral bands recorded by the Landsat system. The location of each of the roughly 50 million pixels composing each image is specified by its matrix coordinates. Also shown is the stored data array for the entire data set prior to clustering. The array contains the seven 1-byte numbers that correspond to the seven different spectral intensities recorded at each pixel location. Thus, prior to clustering, the spectral data for a single Landsat image occupy roughly 350 million bytes of memory. The two components of the stored data array after clustering are shown in (b). The first component is an array consisting of the pixel cluster numbers. (A pixel's cluster number is a 1-byte number that specifies the cluster into which the spectral data for that pixel have been grouped.) The second component is the codebook, or lookup table, which contains the seven spectral values of the centroid of each of the 256 clusters. After clustering, the mean spectral data in the codebook replace all the original spectral data, and the clustered data occupy only about 50 million bytes of memory. As shown in the figure, the cluster number of each pixel links the pixel's coordinates to the appropriate centroid data in the codebook. Those data specify the spectral characteristics of that pixel after clustering, and the figure shows how the clustered data are mapped to an 8-bit-per-pixel color screen. Any combination or mathematical transformation of the seven spectral bands can be used to create the color image. The computer accesses the codebook to find the appropriate spectral values for each pixel.

Figure 6. Using Spectrum to Extract Concepts

The figure shows the window environment provided by Spectrum, a software package developed through a collaboration between the Laboratory and the University of New Mexico. The small window in the lower left-hand corner of the screen shows the entire quarter-scene image. The large center window shows a smaller region at full resolution. This region was selected by placing a box icon within the small window to enclose a particular region. The medium-sized window on the left is a magnification of a region selected by placing a cursor within the center image. Here, the concept-extraction technique has already been applied (as described in the main text). The user drew a polygon within a region identified as an agricultural field, labeled the concept “agriculture,” and selected a shade of bright green as the color for that concept. As a result, all of the agricultural fields in the image (the red-orange regions in Figure 3) have been automatically identified and colored a shade of bright green. The scatter_plot window (upper right) and the class_subform window (lower right) are discussed in Figures 7 and 8.





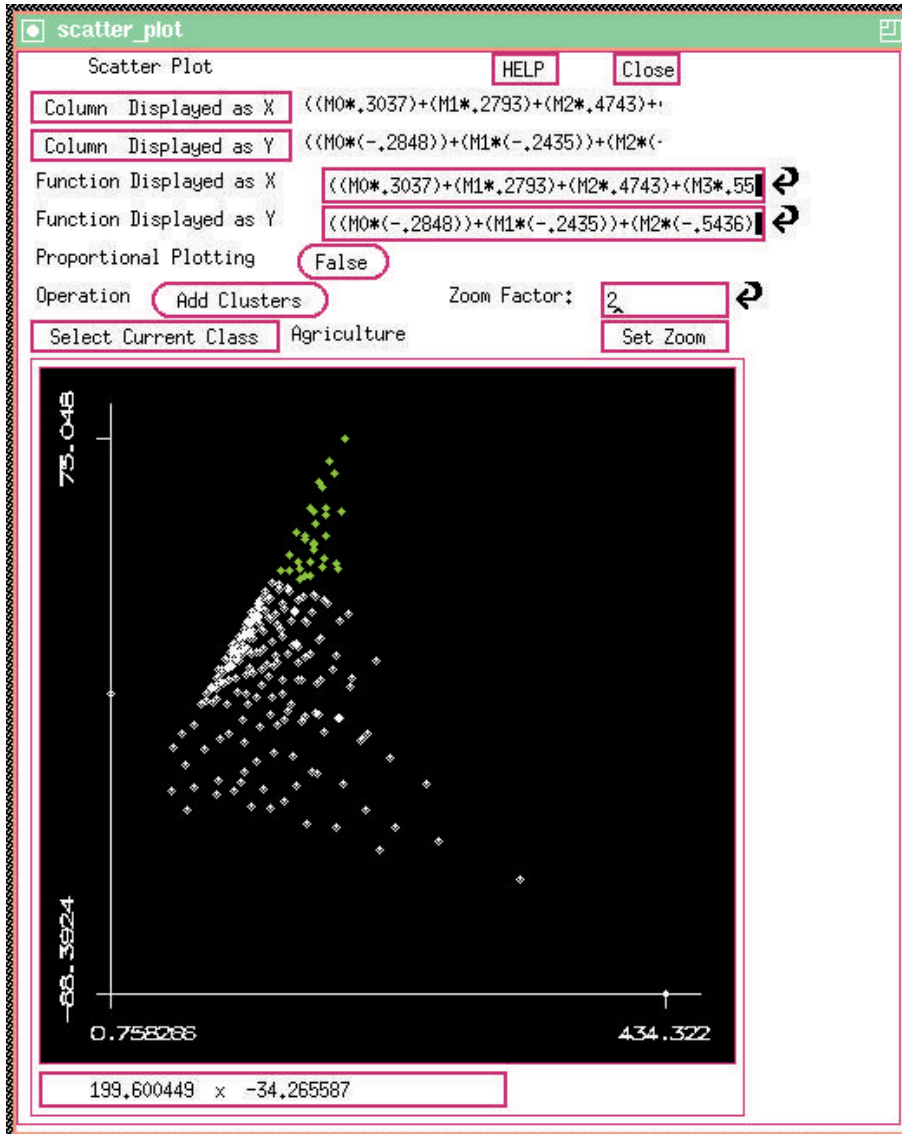


Figure 7. The Scatter_plot Window

The scatter_plot window allows the user to see how clusters within a concept relate to one another as well as to other clusters outside the concept. It is a two-dimensional plot of the 256 cluster centroids. The quantity plotted along each axis is defined by the user as either the intensity recorded in a single spectral band or some function of the intensities in two or more spectral bands. Here the horizontal coordinate (“brightness”) and the vertical coordinate (“greenness”) are linear transformations of the original seven spectral bands. That transformation is known to remote-sensing scientists as the “tasseled-cap” transformation. When the user selects a concept from the menu boxes in the scatter_plot window, all the points representing clusters that have been mapped to that concept are automatically highlighted in the assigned color. Here, for example, the user has selected the concept “agriculture,” which is assigned a shade of bright green, so the points representing centroids whose clusters have been defined as part of the concept “agriculture” are colored bright green. The scatter_plot window reveals points close to those that are bright green and so represent other clusters whose brightness and greenness values fall near those already assigned to the concept but that have not themselves been assigned. Such clusters are also likely to represent agricultural fields, and the user can experiment by adding them to the concept and then examining the resulting mapping.

data stored for each pixel has been reduced from 7 bytes to 1 byte (the pixel’s 1-byte cluster designation), a clustered image can be transmitted seven times faster than the original Landsat image. In addition, the great reduction in the quantity of the spectral data dramatically increases the speed of analysis. For example, calculating the “vegetation vigor” (which is a standard remote-sensing measure derived from the third and fourth spectral bands) of a 12-million-pixel-image from clustered data requires only 768 operations—three operations for each of the 256 clusters. Performing the same calculation on the original data requires 36 million operations, or three for each pixel.

Extracting concepts with Spectrum software. After the clustering algorithm reduces the dataset to a manageable size and level of detail, an interactive data-analysis tool called Spectrum is used to interpret the image. The Spectrum software package was developed through a collaboration between the Laboratory and the University of New Mexico. When using Spectrum, the human expert first identifies an example of a concept, and then the computer takes over and automatically identifies additional occurrences of that concept.

Figure 6 illustrates the interactive use of Spectrum on clustered Landsat data. The user simultaneously views the entire image as well as portions of it at two levels of magnification. To identify a concept of interest, the user simply draws a polygon enclosing a region of the image in one of the magnified views that corresponds, in his or her expert opinion, to that concept. This opinion may be based on the color or colors within the region, the intensity values in other bands of the spectrum, the region’s shape, and/or the position

of the region with respect to other regions. For example, a region used to exemplify the concept “highway” would be gray in color, relatively warm in the thermal infrared band, narrow and relatively straight in shape, and might connect urban areas.

Once the user has drawn a polygon around a region that exemplifies a concept of interest, a category-labeling window appears on the screen. When the user enters the name of the concept, Spectrum automatically defines that concept as the set of all the spectral clusters that are associated with the pixels in the polygon. From the legend-control window, the user then selects a color to represent the concept. Spectrum will then automatically and instantly update the color of all pixels in the image that exemplify any of the spectral clusters that compose the concept. In the image shown in Figure 6, for example, the expert has identified and enclosed a portion of an agricultural region. Suppose that the region is composed of pixels that exemplify, or have been linked through clustering to, spectral clusters 20, 22, 36, and 48. As soon as the enclosed region is labeled “agriculture” and colored, say, bright green, clusters 20, 22, 36, and 48 are labeled “agriculture” and associated with the concept-specific shade of bright green. The color of every pixel in the entire image that exemplifies these clusters is then automatically updated to bright green. In other words, the concept “agriculture” is mapped onto the image.

After the computerized mapping is complete, the user can magnify and examine various regions to which the concept has been mapped and determine whether those regions do indeed exemplify the concept. Some fine tuning may be required. That is, some spectral clusters may be added or removed from the concept definition so

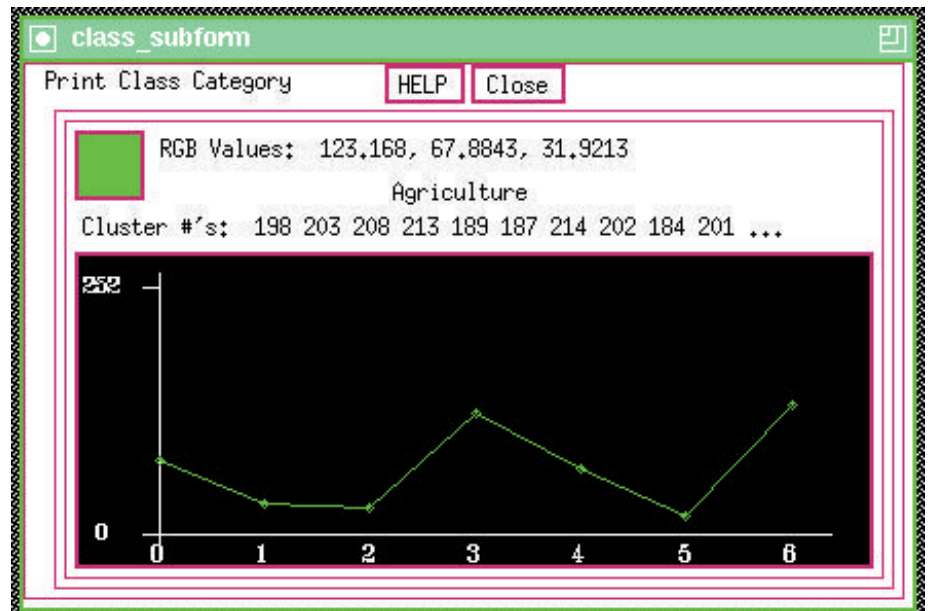


Figure 8. The Class_subform Window

This window plots the relative spectral intensities for any selected pixel. Users who are familiar with spectral data find that such a plot is a helpful addition to the visual information in the magnified images. Here, the spectral characteristics from a pixel in one of the bright green agricultural regions has been selected. Healthy vegetation is generally highly reflective in the infrared region of the spectrum. And, indeed, the plot shows that the infrared intensity measured for this pixel is relatively high. The user might gain added insights by comparing the shape of this plot to that produced by selecting pixels from other regions.

that the mapping more accurately identifies examples of the concept.

Sometimes the relevant concepts are more abstract and apply to regions with quite different spectral characteristics. For example, suppose an analyst is interested in determining what percentage of a given image is used for agriculture. To design the concept “agricultural field,” the expert might draw two polygons, one around a field with growing crops and the other around a fallow field. The analyst would include both those regions in defining a single concept labeled “agriField” and both would be assigned a single color, say, dark blue. This concept unifies regions that are linked not by their spectral characteristics but rather by the more abstract

idea of land use. When both fallow and planted fields identified as “agri-Fields,” quantitative queries such as “what percentage of the image is used for agriculture?” can be posed and answered.

Figures 7 and 8 show two Spectrum windows used for displaying and comparing the spectral data associated with particular pixels, clusters, or concepts. The scatter_plot and class_subform windows provide the user with quantitative representations of the data that are useful tools for designing concepts and evaluating mapping results.

Application of the concept-extraction method to Landsat images has eliminated many of the problems associated with traditional analysis techniques.

The analyst need identify and label only one or a few examples of a concept, and the computer identifies all other examples. That division of labor, combined with the computational efficiency of clustered data, reduces the time required for analysis of an image from several weeks to two or three hours and also decreases the amount of time required to train new analysts. In addition, the data can be displayed on an inexpensive color screen. These advantages have sparked interest in our method from the United States Geological Survey, NASA, and the U.S. Army.

When applied to Landsat data, our concept-extraction technique has also proved capable of achieving the fundamental goal of data mining—finding human concepts amidst huge amounts of data. Obviously, the method would fail if the concepts were poorly separated. For example, our purpose would be subverted if we were to define “agriculture” to include clusters 17, 36, and 45, only to find many pixels of cluster 45 in the middle of the ocean. Such problems do arise occasionally, but the vast majority of cluster assignments are unambiguous. The key to our success lies in keeping clusters “fine-grained” by generating many more clusters than the number of concepts we want to consider.

Concept Extraction and CT Lung Scans

About LAM disease. In collaboration with Dr. John Newell at the National Jewish Center for Respiratory Illnesses in Denver, we are also applying our concept-extraction technique to the analysis of computed-tomography (CT or CAT) scans. Our effort has focused on the scans from women afflicted with lymphangioleiomyomatosis, or LAM disease (see Figure 9). This rare dis-



Figure 9. Computed-Tomography Lung Scan

Computed tomography is a method of recording two-dimensional x-ray images of an internal body structure. An incoming x-ray beam is absorbed by the lung tissue as well as the tissue and bone surrounding the lung. The intensities of the transmitted x rays are recorded by an array of charge counters analogous to those found in CCD cameras. Those intensities are recorded, processed, and reconstructed by a computer to form the image on a video display unit. The CT scan above shows a transverse slice of the left lung of a woman afflicted with a moderate case of LAM disease. Both the spinal column (lower right) and the sternum (upper right) are visible in cross-sectional views. The scan shows multiple, thin-walled cysts throughout the lung. Several normal air-conducting bronchi are also visible, but they are difficult to distinguish from the cysts (see Figure 11). Both the large number of cysts and variations in cyst size make it difficult to gather quantitative diagnostic data on a routine basis.

ease attacks only women in their child-bearing years. The disease is characterized by cysts, or holes, in lung tissue. Its presence is signaled by profound shortness of breath, and diagnosis is confirmed by open-lung biopsy. The cause of the cysts is not yet fully un-

derstood. They may be a result of defective tissue that breaks down during normal breathing, or they may be indirectly caused by the proliferation of smooth muscle tissue within the lungs, a condition often seen in LAM patients. According to the latter theory, blockage

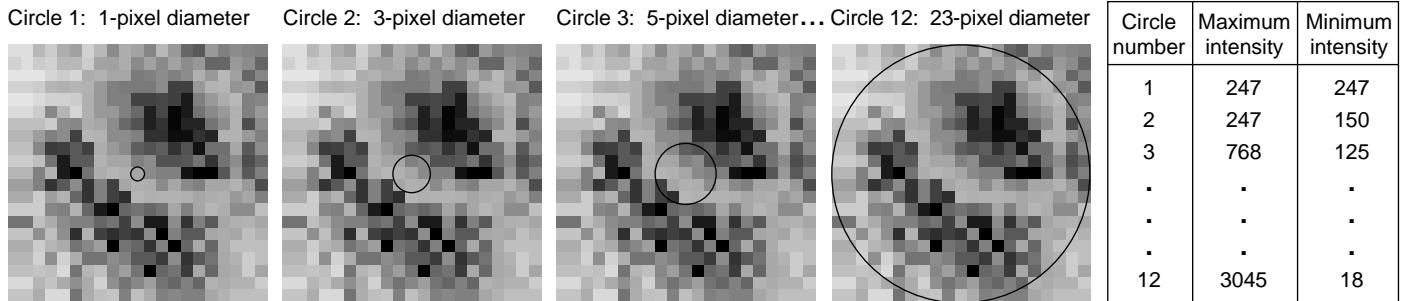


Figure 10. Creating a Quantitative Descriptor

The figure illustrates our method of preparing the CT data for clustering. Before clustering the data, we measure the maximum and minimum intensity of the gray values in twelve concentric circles of increasing diameter centered around each pixel. Only the first three circles and the last circle are illustrated here. The chart shows the maximum and minimum intensity values for each circle. The 24 intensity values are treated as a 24-component vector during clustering. As a result, each pixel is clustered not only in terms of its own gray-scale intensity value but also in terms of the values of the surrounding pixels. The 24-component vectors can be shown to vary according to the tissue density, texture, and local shape of structures in the area surrounding each pixel.

of the airways by the extra muscle tissue causes strain during breathing, which in turn tears the lung tissue. The current therapy involves hormonal manipulation, but unless a lung transplant is performed, LAM disease eventually leads to respiratory failure and death.

LAM disease is notoriously difficult to study. The amount of tissue that can safely be removed in a lung biopsy is too small for research purposes. Entire lungs can be studied when they are removed as part of a transplant operation or after an autopsy; however, the lungs collapse immediately upon removal, and the integrity of the tissue is compromised. Because of these difficulties, LAM researchers are turning to CT data to further their progress in studying this disease.

Previous studies done by “eye-balling” CT scans have indicated that as the disease progresses, patients show increasing numbers of large cysts in their lungs. Researchers are hoping that a more sophisticated analysis of the CT data will reveal more about the origin and progress of the disease and lead to improvements in diagnosis and treatment. We, and Dr. Newell, believe that an application of the concept-extraction

technique may yield significant new results. In his words, concept-extracted CT data could provide a “quantitative, non-invasive, *in vivo* pathology.”

Preparing CT data for concept extraction. Concept extraction is performed on the CT data by using a technique similar to that described above for the Landsat data. The CT data differ, however, in their initial representation and in how they are prepared for clustering. In the case of the Landsat data, “color,” or intensity in seven bands of the electromagnetic spectrum, was the feature that allowed concepts to be differentiated from one another. In contrast, the CT scans record radiation intensity in only one spectral band, namely x rays, and so result in a single black-and-white digital image with pixel intensities ranging on a gray scale from 0 to 4095. These intensity variations alone do not provide enough information to differentiate cysts from normal bronchi because both cysts and bronchi register as “empty,” or black, regions in the CT scan. (Bronchi are hollow regions and cysts are also hollow in the sense that they are holes or tears in the lung tissue.) If Spectrum

were applied directly to the intensity data from the CT scan without any preprocessing, the method would fail. The user might draw a polygon within a cyst to create the concept “cyst.” Spectrum would then identify and map as cysts all pixels with the same intensities as those in the polygon—that is, not only pixels composing additional cysts but also those that compose the bronchi.

To overcome this problem we have defined a quantitative descriptor characterizing the region surrounding each pixel. The descriptor consists of the maximum and minimum intensities of the pixels in each of 12 concentric circles of increasing diameter around each pixel, or 24 intensity measures per pixel (see Figure 10). This 24-component descriptor gives an indication of tissue density, texture, and local shape of structures surrounding a pixel. A special program has been developed to automatically scan the original data and record the 24 components of the descriptor for each pixel. The descriptors provide enough contextual information to avoid the complications outlined above and to produce accurate identification of cysts using Spectrum.

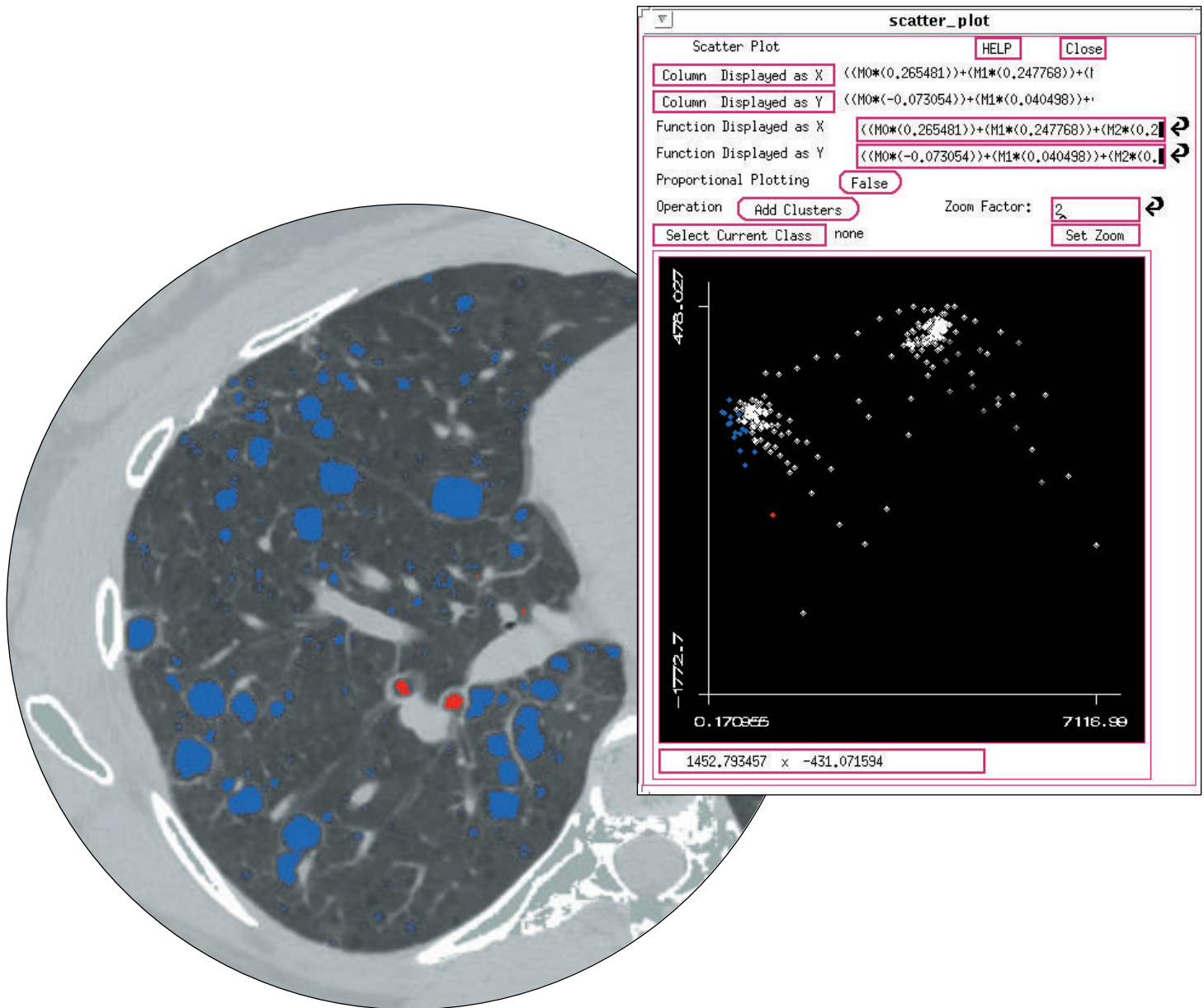


Figure 11. Application of Concept Extraction to a CT Lung Scan

At left is the scan shown in Figure 9 after application of the concept-extraction technique. The intensity data have been pre-processed by using the descriptor defined in Figure 10, and the resulting data have been clustered. Two concepts, “bronchus” and “cyst,” have been defined for the clustered data by expert analysis. Spectrum has been used to map those concepts to the image; red represents “bronchus” and blue represents “cyst.” Two large bronchi are visible near the center of the image, and the scan shows many cysts of various sizes. The descriptor derived from the nested set of circles described in Figure 10 is effective for distinguishing cysts from bronchi primarily because arteries run adjacent to and form a circular pattern around each bronchus. Once the concept-extraction method is applied to the clustered CT data to identify the cysts, researchers can easily carry out quantitative analysis of CT lung scans.

To the right of the CT scan is its scatter plot. The plot shows that the centroids (clusters) corresponding to the concept “cyst” (blue) are well separated from the centroids corresponding to the concept bronchus (red). The separation is achieved by using as the two axes for the plot the first and second principal components of the data: The x-axis is the first principal component, the direction in 24-dimensional space along which the data has maximal variance. The y-axis is the second principal component, the direction perpendicular to the first principal component along which the data shows maximal variance. Note that all the centroids corresponding to lung tissue fall toward the left in this plot, and the centroids corresponding to non-lung tissue fall toward the right. The centroids corresponding to cysts form a more-or-less diagonal line; those toward the upper left correspond to larger cysts and those toward the lower right correspond to smaller cysts.

Concept extraction and automated analysis of CT scans.

Once the 24-component descriptors have been recorded for each pixel in a CT scan, concept extraction proceeds as outlined for the Landsat data. The continuous *k*-means algorithm is used to partition the descriptors into 256 clusters. Spectrum is then used to define four concepts—cyst, bronchus, normal lung tissue, and non-lung material—and to map them to the CT image. Figure 11 shows the lung scan shown in Figure 9 after the first two of these concepts have been mapped to that image. Cysts are shown in blue and bronchi in red. Figure 11 also shows a scatter plot for this image that illustrates the clear separation between the concepts “bronchus” and “cyst” and thus the success of clustering based on the 24-component “contextual” descriptor.

To automate the analysis of concept-mapped data, Los Alamos researchers developed a program that counts the number of cysts of different sizes. This program uses a region-growing algorithm in which a cyst is defined as a group of adjacent pixels each of which has been identified by Spectrum as belonging to the concept “cyst.” The size of the cyst is determined by the number of such adjacent pixels along with the pixel resolution of the CT imagery. The program can also quantify, by means of a central-moment method, other descriptive features of the cyst including eccentricity (deviation from a circular shape) and orientation.

Although our results are preliminary—we have not yet systematically examined a large number of CT scans—the concept-extraction method promises to reveal useful new information about LAM disease. The scans we have analyzed to date support the earlier evidence that more large cysts appear in the later stages of the disease. In addition, computer-assisted techniques have revealed large numbers of

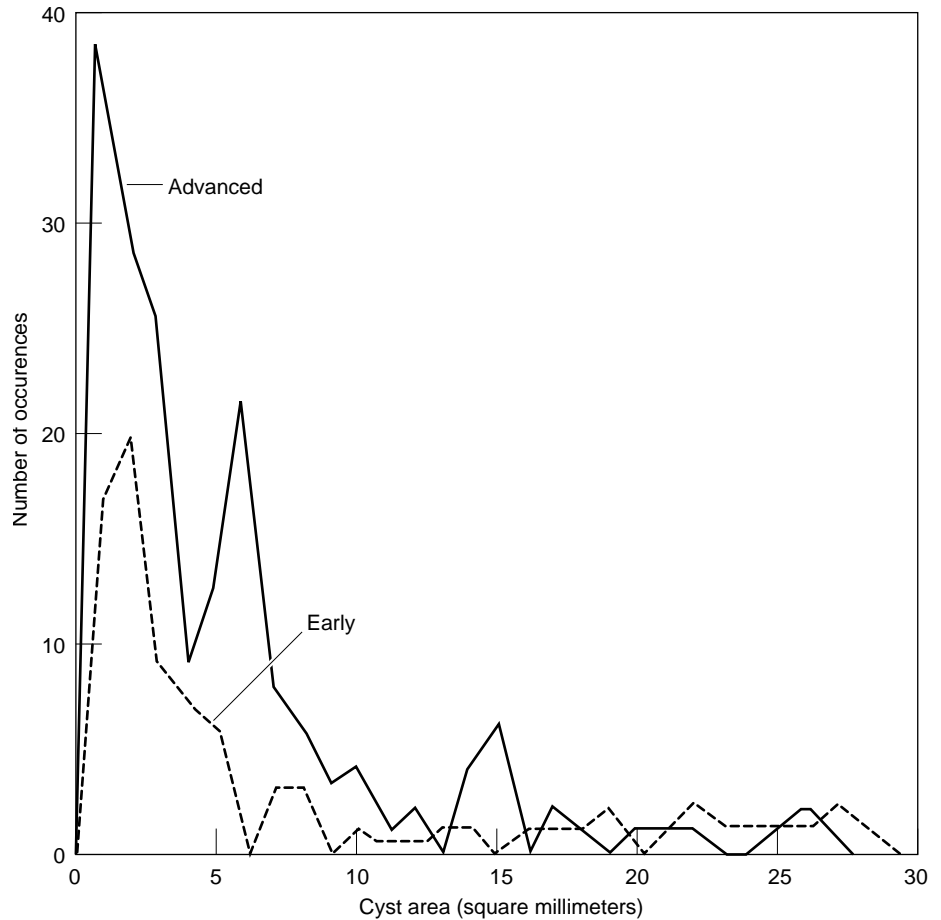


Figure 12. Frequency Distribution of Cyst Sizes

The graphs show the number of cysts of various sizes for two LAM-disease patients. The dotted line represents data from a patient early in the course of the disease, and the solid line, a patient in the later stages. This comparative plot shows that the patient in the later stage not only has more large cysts but also has roughly twice as many small cysts as the patient in the early stage. These results suggest that as the disease progresses, small cysts grow in size and many new small cysts appear. Before the availability of the concept-extraction technique described in the main text, researchers had no efficient way to obtain quantitative measurements of cyst size and number. Now, the progress of the disease in individual patients can be monitored, and researchers can also gain meaningful insights by comparing data from many patients.

small cysts in the later stages of the disease. In other words, not only do small cysts grow larger as the disease progresses but also many new cysts develop. Figure 12 shows graphs of the frequency of different-sized cysts in scans of patients in two stages of the disease. As expected, the patient in the

more advanced stage has more large cysts than the patient in the earlier stage; she also shows a large number of new small cysts. To confirm or refute the proliferation of small cysts in association with the later stages of the disease, the analysis must be repeated on a more substantial number of CT scans.

Clustering and the Continuous k -Means Algorithm

Vance Faber

Many types of data analysis, such as the interpretation of Landsat images discussed in the accompanying article, involve datasets so large that their direct manipulation is impractical. Some method of data compression or consolidation must first be applied to reduce the size of the dataset without losing the essential character of the data. All consolidation methods sacrifice some detail; the most desirable methods are computationally efficient and yield results that are—at least for practical applications—representative of the original data. Here we introduce several widely used algorithms that consolidate data by clustering, or grouping, and then present a new method, the continuous k -means algorithm,* developed at the Laboratory specifically for clustering large datasets.

Clustering involves dividing a set of data points into non-overlapping groups, or clusters, of points, where points in a cluster are “more similar” to one another than to points in other clusters. The term “more similar,” when applied to clustered points, usually means closer by some measure of proximity. When a dataset is clustered, every point is assigned to some cluster, and every cluster can be characterized by a single reference point, usually an average of the points in the cluster. Any particular division of all points in a dataset into clusters is called a partitioning.

One of the most familiar applications of clustering is the classification of plants or animals into distinct groups or species. However, the main purpose of clustering Landsat data is to reduce the size and complexity of the dataset. Data reduction is accomplished by replacing the coordinates of each point in a cluster with the coordinates of that cluster’s reference point. Clustered data require considerably less storage space and can be manipulated more quickly than the original data. The value of a particular clustering method will depend on how closely the reference points represent the data as well as how fast the program runs.

A common example of clustering is the consolidation of a set of students’ test scores, expressed as percentages, into five clusters, one for each letter grade A, B, C, D, and F (see Figure 1). The test scores are the data points, and each cluster’s reference point is the average of the test scores in that cluster. The letter grades can be thought of as symbolic replacements for the numerical reference points.

Test scores are an example of one-dimensional data; each data point represents a single measured quantity. Multidimensional data can include any number of measurable attributes; a biologist might use four attributes of duck bills (four-dimensional data: size, straightness, thickness, and color) to sort a large set of ducks into several species. Each independent characteristic, or measurement, is one dimension. The consolidation of large, multidimensional datasets is the main pur-

* The continuous k -means algorithm is part of a patented application for improving both the processing speed and the appearance of color video displays. The application is commercially available for Macintosh computers under the names *Fast Eddie*, ©1992 and *Planet Color*, ©1993, by Paradigm Concepts, Inc., Santa Fe, NM. This software was developed by Vance Faber, Mark O. Mundt, Jeffrey S. Saltzman, and James M. White.

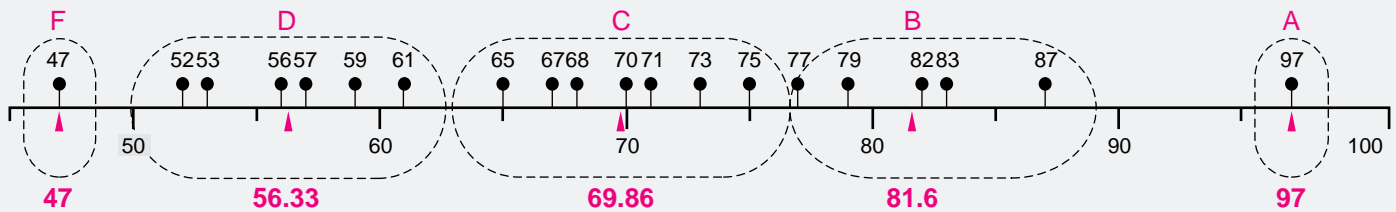


Figure 1. Clustering Test Scores
The figure illustrates an arbitrary partitioning of 20 test scores into 5 non-overlapping clusters (dashed lines), corresponding to 5 letter grades. The reference points (means) are indicated in red.

pose of the field of cluster analysis. We will describe several clustering methods below. In all of these methods the desired number of clusters k is specified beforehand. The reference point z_i for the cluster i is usually the centroid of the cluster. In the case of one-dimensional data, such as the test scores, the centroid is the arithmetic average of the values of the points in a cluster. For multi-dimensional data, where each data point has several components, the centroid will have the same number of components and each component will be the arithmetic average of the corresponding components of all the data points in the cluster.

Perhaps the simplest and oldest automated clustering method is to combine data points into clusters in a pairwise fashion until the points have been condensed into the desired number of clusters; this type of agglomerative algorithm is found in many off-the-shelf statistics packages. Figure 2 illustrates the method applied to the set of test scores given in Figure 1.

There are two major drawbacks to this algorithm. First—and absolutely prohibitive for the analysis of large datasets—the method is computationally inefficient. Each step of the procedure requires calculation of the distance between every possible pair of data points and comparison of all the distances. The second difficulty is connected to a more fundamental problem in cluster analysis: Although the algorithm will always produce the desired number of clusters, the centroids of these clusters may not be particularly representative of the data.

What determines a “good,” or representative, clustering? Consider a single cluster of points along with its centroid or mean. If the data points are tightly clustered around the centroid, the centroid will be representative of all the points in that cluster. The standard measure of the spread of a group of points about its mean is the variance, or the sum of the squares of the distance between each point and the mean. If the data points are close to the mean, the variance will be small. A generalization of the variance, in which the centroid is replaced by a reference point that may or may not be a centroid, is used in cluster analysis to indicate the overall quality of a partitioning; specifically, the error measure E is the sum of all the variances:

$$E = \sum_{i=1}^k \sum_{j=1}^{n_i} \|x_{ij} - z_i\|^2,$$

where x_{ij} is the j th point in the i th cluster, z_i is the reference point of the i th cluster, and n_i is the number of points in that cluster. The notation $\|x_{ij} - z_i\|$ stands for the distance between x_{ij} and z_i . Hence, the error measure E indicates the overall spread of data points about their reference points. To achieve a representative clustering, E should be as small as possible.

The error measure provides an objective method for comparing partitionings as well as a test for eliminating unsuitable partitionings. At present, finding the best

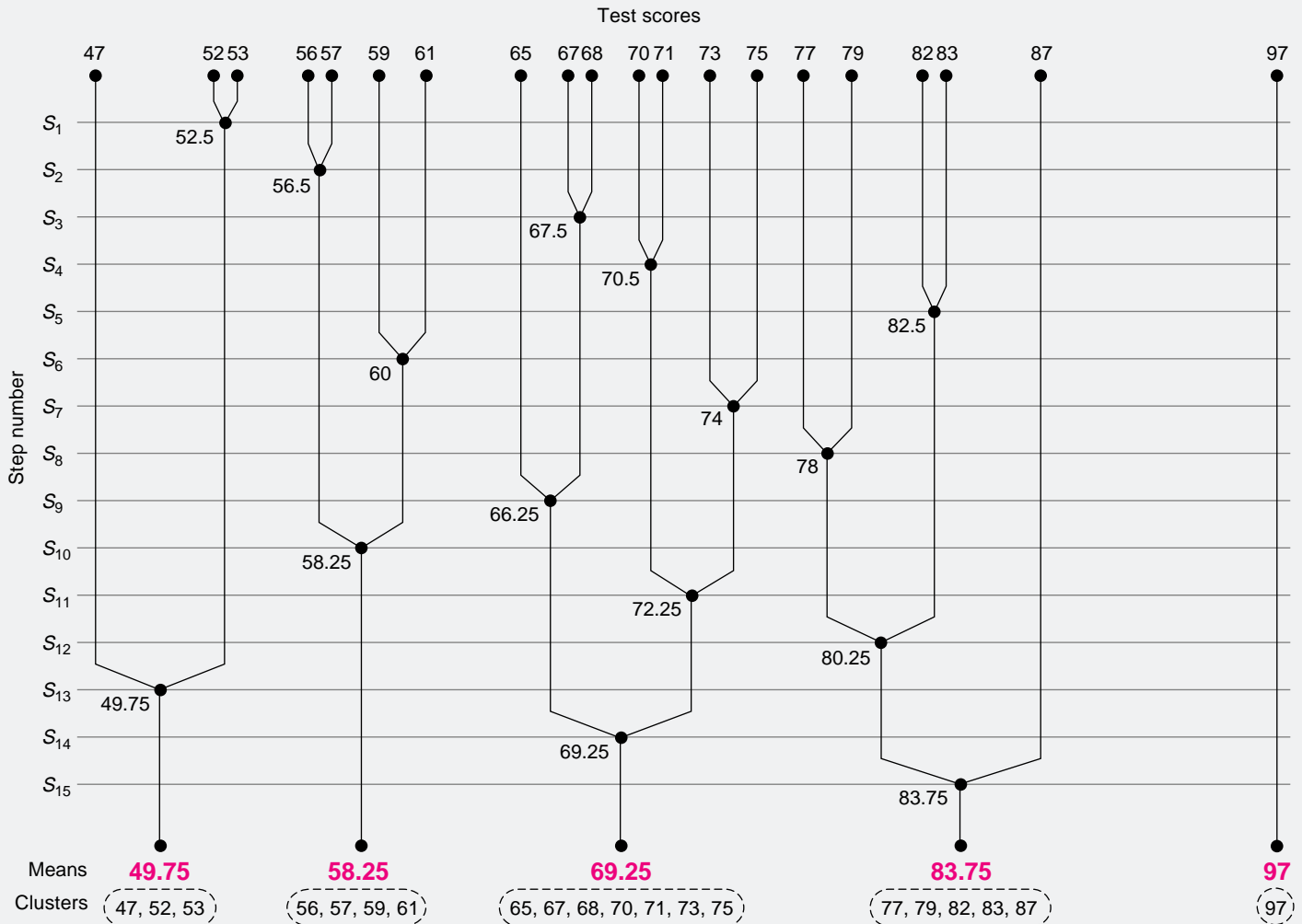


Figure 2. Pairwise Agglomerative Clustering

The figure illustrates the operation of an agglomerative clustering method, in which the 20 test scores of Figure 1 are successively merged by pairs of points and/or pairs of clusters until all the scores are collected into 5 clusters. The steps of the algorithm are shown in the branching of a dendrogram, or tree structure (much like a genealogy). A node, or branch point, indicates the merging of two branches into one, i.e. two data points into one cluster, or two clusters into one larger cluster. The algorithm begins with 20 separate clusters of one point apiece. For the first step in the algorithm, the closest two points (here, scores of 52 and 53) are found and merged into one cluster {52,53}. The two individual points are replaced by a single point equal to the unweighted average of the two points (52.5). The next step repeats this process (find the closest two points, calculate the average, merge the points), but with 19 points and 19 clusters (18 one-point clusters, plus 1 two-point cluster). There will be only one new branch, or merge at each step. Hence, if there is more than one pair of points at the minimum distance, only one pair will be merged at each step. It takes 15 steps to consolidate 20 points into 5 clusters.

partitioning (the clustering most representative of an arbitrary dataset) requires generating all possible combinations of clusters and comparing their error measures. This can be done for small datasets with a few dozen points, but not for large sets—the number of different ways to combine 1 million data points into 256 clusters, for example, is $256^{1,000,000}/256!$, where $256!$ is equal to $256 \times 255 \times 254 \times \cdots \times 2 \times 1$. This number is greater than $10^{2,000,000}$, or 1 followed by 2 million zeros.

When clustering is done for the purpose of data reduction, as in the case of the Landsat images, the goal is not to find the best partitioning. We merely want a reasonable consolidation of N data points into k clusters, and, if necessary, some efficient way to improve the quality of the initial partitioning. For that purpose, there is a family of iterative-partitioning algorithms that is far superior to the agglomerative algorithm described above.

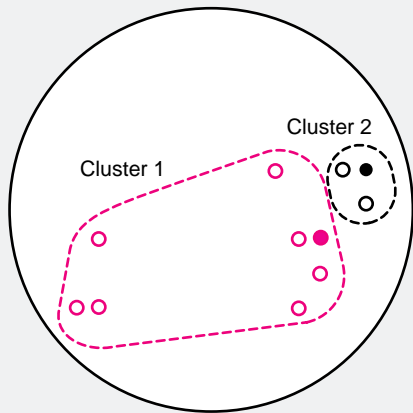
Iterative algorithms begin with a set of k reference points whose initial values are usually chosen by the user. First, the data points are partitioned into k clusters: A data point x becomes a member of cluster i if z_i is the reference point closest to x . The positions of the reference points and the assignment of the data points to clusters are then adjusted during successive iterations. Iterative algorithms are thus similar to fitting routines, which begin with an initial “guess” for each fitted parameter and then optimize their values. Algorithms within this family differ in the details of generating and adjusting the partitions. Three members of this family are discussed here: Lloyd’s algorithm, the standard k -means algorithm, and a continuous k -means algorithm first described in 1967 by J. MacQueen and recently developed for general use at Los Alamos.

Conceptually, Lloyd’s algorithm is the simplest. The initial partitioning is set up as described above: All the data points are partitioned into k clusters by assigning each point to the cluster of the closest reference point. Adjustments are made by calculating the centroid for each of those clusters and then using those centroids as reference points for the next partitioning of all the data points. It can be proved that a local minimum of the error measure E corresponds to a “centroidal Voronoi” configuration, where each data point is closer to the reference point of its cluster than to any other reference point, and each reference point is the centroid of its cluster. The purpose of the iteration is to move the partition closer to this configuration and thus to approach a local minimum for E .

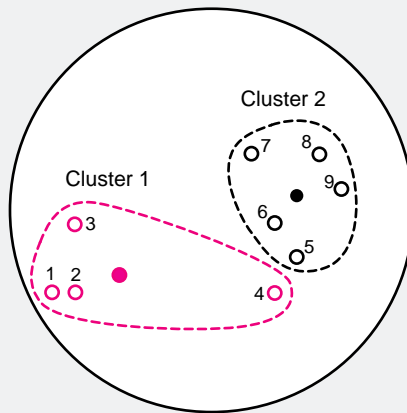
For Lloyd’s and other iterative algorithms, improvement of the partitioning and convergence of the error measure E to a local minimum is often quite fast—even when the initial reference points are badly chosen. However, unlike guesses for parameters in simple fitting routines, slightly different initial partitionings generally do not produce the same set of final clusters. A final partitioning will be better than the initial choice, but it will not necessarily be the best possible partitioning. For many applications, this is not a significant problem. For example, the differences between Landsat images made from the original data and those made from the clustered data are seldom visible even to trained analysts, so small differences in the clustered data are even less important. In such cases, the judgment of the analyst is the best guide as to whether a clustering method yields reasonable results.

(a) Setup:

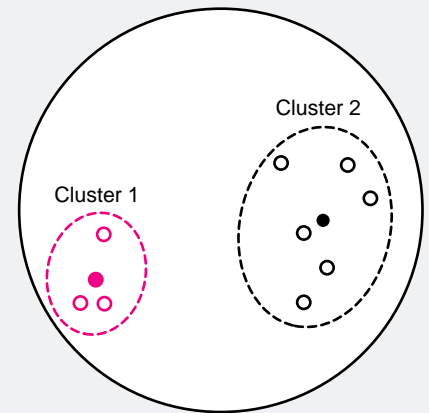
Reference point 1 (filled red circle) and reference point 2 (filled black circle) are chosen arbitrarily. All data points (open circles) are then partitioned into two clusters: each data point is assigned to cluster 1 or cluster 2, depending on whether the data point is closer to reference point 1 or 2, respectively.

**(b) Results of first iteration:**

Next each reference point is moved to the centroid of its cluster. Then each data point is considered in the sequence shown. If the reference point closest to the data point belongs to the other cluster, the data point is reassigned to that other cluster, and both cluster centroids are recomputed.

**(c) Results of second iteration:**

During the second iteration, the process in Figure 3(b) is performed again for every data point. The partition shown above is stable; it will not change for any further iteration.

**Figure 3. Clustering by the Standard k -Means Algorithm**

The diagrams show results during two iterations in the partitioning of nine two-dimensional data points into two well-separated clusters, using the standard k -means algorithm. Points in cluster 1 are shown in red, points in cluster 2 are shown in black; data points are denoted by open circles and reference points by filled circles. Clusters are indicated by dashed lines. Note that the iteration converges quickly to the correct clustering, even for this bad initial choice of the two reference points.

The standard k -means algorithm differs from Lloyd's in its more efficient use of information at every step. The setup for both algorithms is the same: Reference points are chosen and all the data points are assigned to clusters. As with Lloyd's, the k -means algorithm then uses the cluster centroids as reference points in subsequent partitionings—but the centroids are adjusted both during and after each partitioning. For data point x in cluster i , if the centroid z_i is the nearest reference point, no adjustments are made and the algorithm proceeds to the next data point. However, if the centroid z_j of the cluster j is the reference point closest to data point x , then x is reassigned to cluster j , the centroids of the “losing” cluster i (minus point x) and the “gaining” cluster j (plus point x) are recomputed, and the reference points z_i and z_j are moved to their new centroids. After each step, every one of the k reference points is a centroid, or mean, hence the name “ k -means.” An example of clustering using the standard k -mean algorithm is shown in Figure 3.

There are a number of variants of the k -means algorithm. In some versions, the error measure E is evaluated at each step, and a data point is reassigned to a different cluster only if that reassignment decreases E . In MacQueen's original paper on the k -means method, the centroid update (assign data point to cluster, recompute the centroid, move the reference point to the centroid) is applied at each step in the initial partitioning, as well as during the iterations. In all of these cases, the standard k -means algorithm requires about the same amount of computation for a single pass through all the data points, or one iteration, as does Lloyd's algorithm. However, the k -means algorithm, because it constantly updates the clusters, is unlikely to require as many iterations as the less efficient Lloyd's algorithm and is therefore considerably faster.

The Continuous k -Means Algorithm

The continuous k -means algorithm is faster than the standard version and thus extends the size of the datasets that can be clustered. It differs from the standard version in how the initial reference points are chosen and how data points are selected for the updating process.

In the standard algorithm the initial reference points are chosen more or less arbitrarily. In the continuous algorithm reference points are chosen as a random sample from the whole population of data points. If the sample is sufficiently large, the distribution of these initial reference points should reflect the distribution of points in the entire set. If the whole set of points is densest in Region 7, for example, then the sample should also be densest in Region 7. When this process is applied to Landsat data, it effectively puts more cluster centroids (and the best color resolution) where there are more data points.

Another difference between the standard and continuous k -means algorithms is the way the data points are treated. During each complete iteration, the standard algorithm examines all the data points in sequence. In contrast, the continuous algorithm examines only a random sample of data points. If the dataset is very large and the sample is representative of the dataset, the algorithm should converge much more quickly than an algorithm that examines every point in sequence. In fact, the continuous algorithm adopts MacQueen's method of updating the centroids during the initial partitioning, when the data points are first assigned to clusters. Convergence is usually fast enough so that a second pass through the data points is not needed.

From a theoretical perspective, random sampling represents a return to MacQueen's original concept of the algorithm as a method of clustering data over a continuous space. In his formulation, the error measure E_i for each region R_i is given by

$$E_i = \int_{x \in R_i} \rho(x) \|x - z_i\|^2 dx,$$

where $\rho(x)$ is the probability density function, a continuous function defined over the space, and the total error measure E is given by the sum of the E_i 's. In MacQueen's concept of the algorithm, a very large set of discrete data points can be thought of as a large sample—and thus a good estimate—of the continuous probability density $\rho(x)$. It then becomes apparent that a random sample of the dataset can also be a good estimate of $\rho(x)$. Such a sample yields a representative set of cluster centroids and a reasonable estimate of the error measure without using all the points in the original dataset.

These modifications to the standard algorithm greatly accelerate the clustering process. Since both the reference points and the data points for the updates are chosen by random sampling, more reference points will be found in the densest regions of the dataset and the reference points will be updated by data points in the most critical regions. In addition, the initial reference points are already members of the dataset and, as such, require fewer updates. Therefore, even when applied to a large dataset, the algorithm normally converges to a solution after only a small fraction (10 to 15 percent) of the total points have been examined. This rapid convergence distinguishes the continuous k -means from less efficient algorithms. Clustering with the continuous k -means algorithm is about ten times faster than clustering with Lloyd's algorithm.

The computer time can be further reduced by making the individual steps in the algorithm more efficient. A substantial fraction of the computation time required by any of these clustering algorithms is typically spent in finding the reference point closest to a particular data point. In a “brute-force” method, the distances from a given data point to all of the reference points must be calculated and compared. More elegant methods of “point location” avoid much of this time-consuming process by reducing the number of reference points that must be considered—but some computational time must be spent to create data structures. Such structures range from particular orderings of reference points, to “trees” in which reference points are organized into categories. A tree structure allows one to eliminate entire categories of reference points from the distance calculations. The continuous k -means algorithm uses a tree method to cluster three-dimensional data, such as pixel colors on a video screen. When applied to seven-dimensional Landsat data, the algorithm uses single-axis boundarizing, which orders the reference points along the direction of maximum variation. In either method only a few points need be considered when calculating and comparing distances. The choice of a particular method will depend on the number of dimensions of the dataset.

Two features of the continuous k -means algorithm—convergence to a feasible group of reference points after very few updates and greatly reduced computer time per update—are highly desirable for any clustering algorithm. In fact, such features are crucial for consolidating and analyzing very large datasets such as those discussed in the accompanying article. □

Further Reading

James M. White, Vance Faber, and Jeffrey S. Saltzman. 1992. Digital color representation. U.S. Patent Number 5,130,701.

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* IT-28: 129–137.

Edward Forgy. 1965. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications, *Biometrics* 21: 768.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I, Statistics*. Edited by Lucien M. Le Cam and Jerzy Neyman. University of California Press.

Jerome H. Friedman, Forest Baskett, and Leonard J. Shustek. 1975. An algorithm for finding nearest neighbors. *IEEE Transactions on Computers* C-24: 1000-1006. [Single-axis boundarizing, dimensionality.]

Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3: 209–226. [Tree methods.]

Helmuth Späth. 1980. *Cluster Analysis Algorithms for Data Reduction and Classification of Objects*. Halsted Press.

Anil K. Jain and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall.

The biography of Vance Faber appears on page 149.

Concept Extraction Applied to Text Analysis of Medical Records

Challenges in analyzing textual data. Textual datasets are very different from the digital-image data to which we originally applied our concept-extraction technique. Since the technique had been conceived in a general way and was not specific to any particular kind of data, we felt confident that our method could also be applied in the analysis of textual data. Our attempts are in their early stages and several details still need to be worked out, but our progress to date has served to reinforce our confidence in the approach. The textual data consist of transcripts of physicians' notes on patient visits to a large health-maintenance organization (HMO). Each transcription is termed a document, and each document is a unique data element. The sample we analyzed contains 142,475 such documents—the entire HMO output for a recent ten-month period. To ensure privacy, the patients' names and medical-record numbers are replaced by randomly chosen pseudonyms, access to the data is strictly controlled, and all identifying information is altered in publicly distributed documents.

Current methods of analysis. The current method of extracting information from transcribed patient records is to bind printed copies of all documents related to a single patient in a medical record folder and deliver that folder to the physician whenever the patient is treated at the HMO. The physician then reviews the documents and extracts information deemed helpful for determining a course of treatment. On occasion, epidemiologists or medical researchers will collect a small sample of related cases and study the patients'

records in an attempt to answer general medical questions related to treatments and outcomes.

Obviously, these two extraction methods are limited by the ability of highly-trained human readers to accurately and comprehensively read the documents. A dataset of over 100,000 such documents cannot be examined with current methods, yet the document set as a whole probably contains a considerable amount of useful information, such as answers to questions like: What diseases are most common in this patient population? Is aspirin an effective treatment for cystitis? Is the rate of attempted suicide higher among teenagers or adults? Answering such questions requires a computerized method of examining large datasets of textual documents and extracting conceptual information.

It may seem that the solution is not to train computers to analyze text but rather to train physicians to record their patient encounters in a structured way so it will be easier to analyze the data—"fill out the form!" Many attempts have been made to standardize medical-data recording, but all have been successfully resisted by physicians. They argue that only free text can adequately capture the inherent ambiguity of the concepts involved in a medical encounter. Consider the concept "pain." Pain may be constant or intermittent or associated with a specific movement, and there are many different ways of describing it: sharp, dull aching, throbbing, and so on. Physicians are keenly aware of the inherent ambiguity of conceptual labels and have insisted on maintaining their freedom to record encounters in a free-text format. Therefore, if the information in the dataset is to be made more generally useful, we must solve the difficult problem of extracting it from documents written in a free-text format.

Choosing a quantitative descriptor and measure of "closeness." Before the dataset can be clustered, we must choose a quantitative descriptor for each data element. The most obvious numerical representation of text is a descriptor indicating the frequencies of all words in each document. However, since the number of words encountered in medical records is very large (hundreds of thousands) and is constantly increasing, we wanted to avoid the problem of managing such huge descriptors. Choosing a limited set of words based on expert recommendations might reduce the difficulties, but even an expert might overlook terms of great significance. In addition, parsing ASCII text into "words" presents problems with respect to handling compound words, abbreviations, homonyms, hyphenations, spelling errors, morphological alternatives (such as *nausea* and *nauseous*), and so on.

Our solution to these problems is the use of a quantitative representation of each document that is based on words but avoids using the words themselves. The method, called character-trigram representation, represents each document as the set of frequencies of all three-letter sequences (trigrams) in the document. To create this representation, all non-alphabetic characters in the document, including spaces, are removed and all letters are reduced to lower case. A three-letter window is then run over the document and the frequencies of the different trigrams is tallied. The phrase "Prozac 20 mg daily," for example, yields one instance each of trigrams *pro*, *roz*, *oza*, *zac*, *acm*, *cmg*, *mgd*, *gda*, *dai*, *ail*, and *ily*. Trigrams—or more generally, *n*-grams of varying lengths—have been used successfully in text applications ranging from spell checking to language identification.

The main advantage of the trigram approach is that the number of trigrams

that must be tallied is relatively small—about 14,000. [The total number of trigrams that can be found in words written in the Latin alphabet is $17,576$ (26^3), but many of the possible trigrams, including, for example, *bbb*, and *yzv*, are seldom if ever encountered.] So the use of trigrams brings the clustering problem down to a size that our continuous *k*-means algorithm can easily handle without our having to select a list of words beforehand. In addition, trigrams can be tallied without any of the difficult preprocessing that a tally of words would require. Variations in word form due to spelling errors and morphological variations “wash out” because trigrams provide a shared representation for word variants. For example, morphological variants such as *congested* and *congestion* overlap and share the trigrams *con*, *ong*, *nge*, *ges*, and *est*. Likewise, *telephone* and *telepbone* (the latter the result of a plausible optical-character-recognition error) share trigrams *tel*, *ele*, *lep*, and *one*.

The next task before clustering is the selection of a quantitative measure of the similarity, or “closeness,” of two sets of trigram frequencies—which provides a measure of the “closeness” of two documents. We chose to consider the sets as vectors in a space of about 14,000 dimensions, one dimension for each trigram, and to measure the distance in terms of the cosine of the angle between the trigram vectors for the two documents. The angle between two vectors does not depend on their lengths, so this measure allows the unbiased comparison of documents of different lengths. We calculate the cosine according to a standard formula of vector analysis. If the cosine is equal to 1 (meaning the vectors are collinear), the documents have identical trigram-frequency distributions and are very similar if not identical. If the cosine is 0 (meaning the vectors are perpendicu-

lar), then the documents have no trigrams in common, and thus no words in common. More complicated measures, which take into account the relative variability of specific trigrams across the dataset, could be constructed and may prove useful. An obvious enhancement would be to weight common trigrams such as *the* and *and* less heavily than others. In our early stage of exploration, however, we have chosen the simplest approach.

Testing the method. Before attempting to cluster the documents, we wanted to be certain that documents that are close to one another according to our cosine measure are also close to one another in the sense that they relate to the same topic. To test our method we selected a document that was clearly about headaches and searched the dataset for the closest, or most similar, documents according to the trigram-frequency descriptor. Of the ten closest documents, eight related to headaches, one to a numb foot, and one to dizziness. Those documents that were not related to headaches were, however, generated by the same doctor and written on the same day as the document relating to headaches selected as our reference document.

These results were compared with results obtained by using a popular text retrieval tool (WAIS), which works by counting the number of words in common between two documents. We found that WAIS returned one document about headaches, the two non-headache documents our method also found, and seven other documents that were about a variety of medical topics including stroke, facial tic, and back pain. Given this favorable comparison, we proceeded with clustering under the assumption that similarity in “trigram space” indicates similarity in “topic space.” It is important to note that the

documents have different meanings—each describes very different situations. The similarity resides in the topic of the documents. Since they contain similar root words describing that topic, they produce similar trigram distributions. The sentences “she is healthy” and “he is unhealthy” have different meanings, but both are about the same topic—health—and they share a similar trigram distribution.

Clustering the documents. We chose to create 1000 clusters, and the results yielded various types of clusters. One cluster is very large, containing over 900 documents. All members of this cluster appear to be documents that concern vague headaches. Without any further analysis, we can reasonably conclude that headaches are a very common symptom prompting numerous visits to the HMO; visits relating to headaches should be considered seriously when planning resource allocations. It is very likely, however, that other clusters also relate to headaches, and it is necessary to fuse such clusters into a single conceptual category before meaningful quantitative estimates can be made. It is important to note, however, that the clustering process produces valuable results even without fusing clusters. Clustering of a very large dataset greatly reduces the time needed to locate similar documents, since one can first find the cluster to which the reference document belongs and then search only the members of that and nearby clusters for similar documents. This procedure is much faster than using an algorithm that considers all documents as candidates, but it yields virtually identical results.

Our results yielded a second type of cluster, a singleton, which contains only one document. Any document found in a singleton usually relates to some unique topic or is simply some sort of

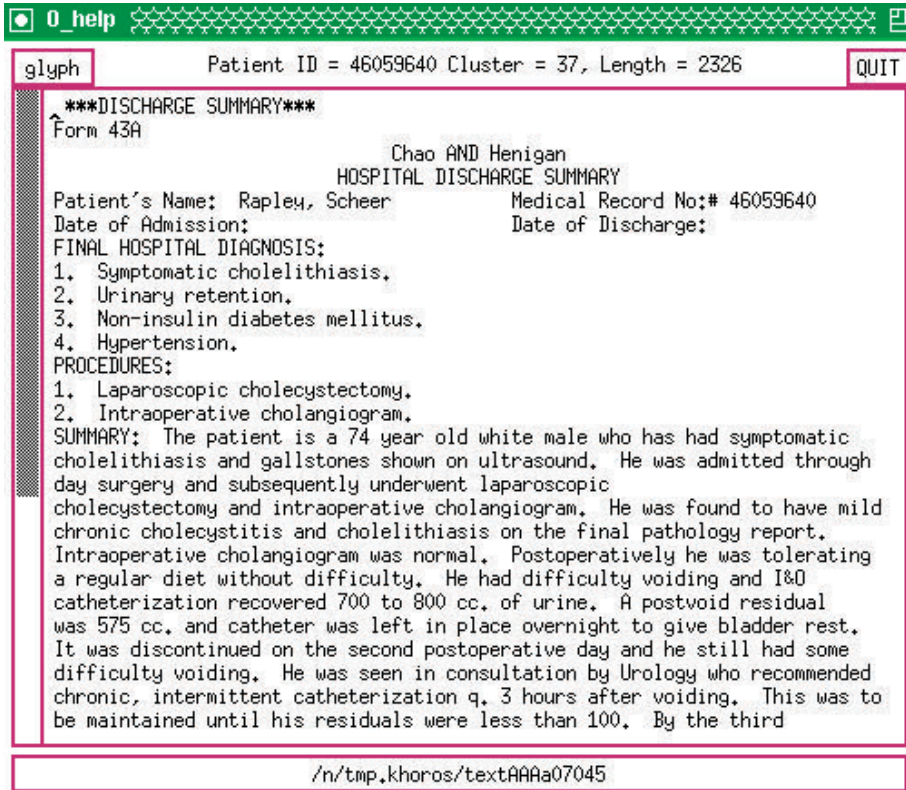
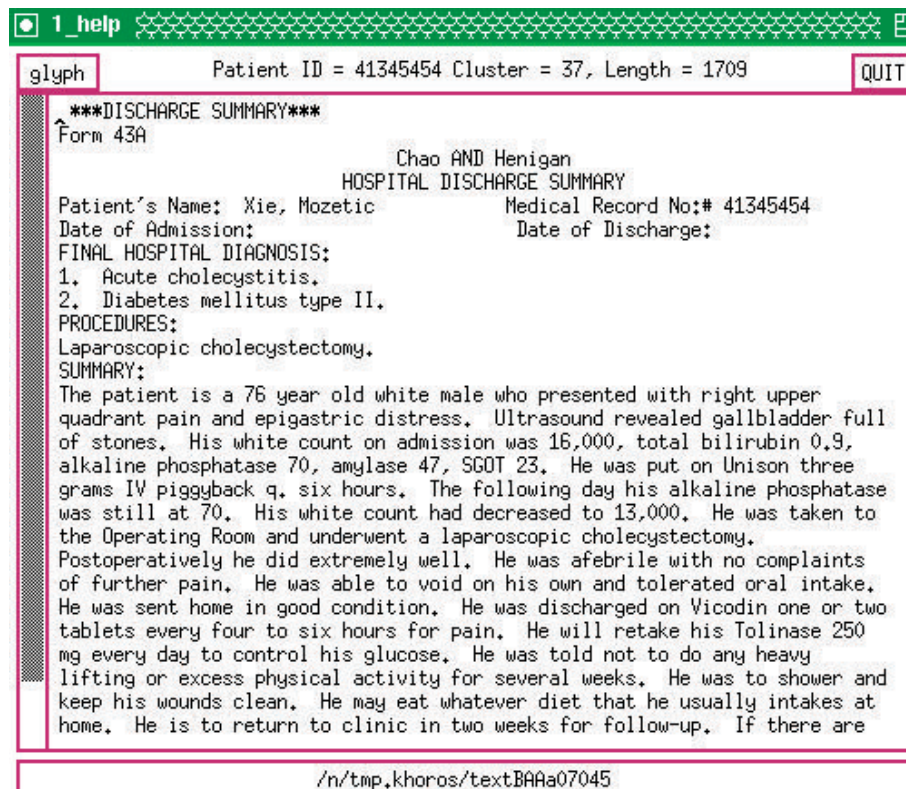


Figure 13. Example of a Two - Member Cluster

The figure shows an example of the type of document (transcripts from doctors' notes on-patient visits) we have been working with in our textual dataset. These two documents represent a doubleton cluster, a cluster containing only two documents that are so much more similar to each other than to the other documents that the algorithm automatically separated them into their own cluster. Here, both documents relate to cases of laparoscopic cholestectomy, both patients are white males in their mid-70s, and the documents share several other medical details in common. Since laparoscopic cholestectomy is a fairly common procedure, it seems likely that there are more instances of this procedure in other clusters.



error in the transcription process. A third type of cluster is the doubleton, which contains two very similar documents (see Figure 13).

Fusing related clusters. Any closely related clusters must be fused before any quantitative information about a given condition, procedure, or therapy can be obtained, and we are currently trying to develop an efficient method of fusing such clusters. The underlying difficulty is that people are best able to evaluate the clusters based on words, but the clusters were created from their trigram content. We are making some progress in finding a way to map the trigram distributions back to a meaningful list of words so that the expert can fuse seemingly disjoint clusters into conceptually relevant categories.

The basic idea is to rank the words in each document according to how closely they mirror the cluster centroid. Words that contain trigrams that occur frequently in the cluster centroid are considered to be more indicative of that centroid than those containing less frequently occurring trigrams. To begin, we assign a weight for each possible trigram by taking the normalized frequency of each trigram in the cluster centroid. We then determine the importance of each letter in every document in the cluster by adding the weights of the three trigrams to which that letter belongs, given its left and right context. A word-weight is constructed by taking the average letter-weight for the entire word. A ranked word-list can then be compiled. A “stop” list is applied to eliminate non-content words, and a stemming algorithm is applied so that redundancy in the ranked list is reduced by eliminating multiple occurrences of words that are simple variants of the same stem (*operate*, *operates*, *operating*, etc.) Finally, a list of the ten most important

(that is, the ten most heavily weighted) words is created for each cluster.

The top ten members of this list for the two-member cluster in Figure 13 are: *tolerate*, *operate*, *intake*, *postoperative*, *lifting*, *tolinase*, *white*, *intraoperative*, *laparoscopic*, and *stone*. This list catalogues keywords describing the most important elements of the centroid for this cluster. Such lists are useful for retrieving documents on the basis of keyword searches. The lists also represent practical, quick-reference descriptions of the contents of the various clusters, and the user can decide whether to read particular documents by examining these lists.

We are also planning a user-interface to help the expert decide which clusters should be fused. It is impractical to expect an expert to compare and evaluate all the lists for the 1000 clusters—even glancing at such lists is an onerous task. We plan to provide an interface that presents a small group of similar lists (similar according to a measure of the co-occurrence of either trigrams or words), prompts the expert to decide whether the lists should be fused to a concept, and then asks what the name and descriptive words for that concept should be. Then, on the basis of the concept-label words, other possible candidate lists will be displayed and the process iterated until the expert is satisfied. At that point quantitative estimates of documents associated with each labeled concept can be made.

Enhancing the Method and Developing New Applications

The research reported in this paper is part of an ongoing, long-term effort. We are working to develop new techniques that will improve the existing technology. In the area of digital images, for example, so far we have clus-

tered on the basis of either the spectral intensities associated with each pixel (as with the Landsat data) or the texture surrounding a pixel (as with the CT data). If we were able to combine both types of information, the cluster analysis might result in more subtle distinctions, such as trees in a suburban neighborhood versus trees in a forest. We are also experimenting with methods of generalizing the concepts extracted from an image of one area to images of other areas. Such a technique would allow the analyst to identify a concept, say, “deciduous forest,” in one or two training images and then have any deciduous forest automatically identified in other images. Because every image produces a unique set of clusters, the success of this technique may hinge on our ability to map concepts to clusters identified by their relationship to other clusters, rather than by their specific centroid values.

We also hope to extend our methods to other data domains. Within the area of digital-image processing, we are exploring the analysis of x-ray images, and within the area of text processing, we are working with datasets pertaining to arms control and physics research. A logical new domain to tackle is that of one-dimensional signal analysis so that sound or sonar spectra can be clustered and mapped to concepts. A more challenging extension of our method would be to process data in which several different types of information—character, categorical, scalar—are recorded for each event in a dataset. For example, an intrusion detection program might analyze computer audit records containing the name of the user and the process (character), the duration of the process (scalar), and the error status of the process (categorical) for each completed process. A representation that combines these different types of information in a form that allows

clustering would, at a minimum, allow more efficient handling of typically huge audit records.

We view our concept-extraction technique as a bridge between imprecise human concepts and the world of physical, quantitative measurements. The method addresses a major weakness in the field of artificial intelligence: Although machine-learning algorithms enable computers to master real data, they fall short of being capable of what humans think of as intelligent behavior because the results of the learning process do not reflect human concepts. Rather than try to solve this problem directly, we have developed a tool that uses a partial statistical analysis to facilitate the mapping of concepts onto data. Until researchers develop a more fundamental solution that enables computers to discover human-relevant concepts on their own, concept extraction will allow us to make intelligent use of the massive amounts of data already being collected. ■

Further Reading

A. K. Jain and R. C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall.

W. Labov. 1973. The boundaries of words and their meanings. In *New Ways of Analyzing Variation in English*. Georgetown University Press.

P. M. Kelly and J. M. White. Preprocessing remotely-sensed data for efficient analysis and classification. In *SPIE Applications of Artificial Intelligence 1993: Knowledge-Based Systems in Aerospace and Industry*. 1993: 24–30.

Ching Y. Suen. 1979. *N*-gram statistics for natural language understanding and text processing. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1: 164–172.

Roy E. Kimbrell. 1988. Searching for text? Send an *n*-gram! *Byte*. May, 297–312.



Patrick M. Kelly, James M. White, Judith G. Hochberg, Timothy R. Thomas, and Vance Faber

Vance Faber received his B.A. and M.A. degrees in mathematics in 1966 and 1969 and his Ph.D. in combinatorial group theory in 1971 from Washington University in St. Louis. After nine years as a professor at the University of Colorado at Denver, he accepted a position as a staff member in the Laboratory's Computer Research and Applications Group. Faber has been the leader of that group since 1992. He is extremely interested in novel applications of theoretical mathematics to problems that have societal impact.

Judith G. Hochberg received her B.A. in linguistics from Harvard University in 1982 and her M.S. and Ph.D. from Stanford University in 1985 and 1986. She came to the Laboratory in 1989 as a director-funded postdoctoral fellow, and since 1991 she has worked in the Computer Research and Applications Group. Before joining the Laboratory, Hochberg published research on sociolinguistics and child language learning with a particular interest in phonological rules. Hochberg's current research interests include various topics in computational linguistics as well as work in computer security and anomaly detection.

Patrick M. Kelly came to the Laboratory as a graduate research assistant in 1990, and in 1992 he joined the Computer Research and Applications Group as a technical staff member. He has participated in various projects, including Landsat data analysis, automated quantitative analysis of medical imagery, the development of clustering algorithms for massive digital datasets, and digital-image comparison and retrieval. His primary research interests include image processing and pattern recognition. Kelly earned his B.S. in computer engineering from the University of New Mexico in 1990 and his M.S. in electrical engineering from the University of New Mexico, Los Alamos, in 1992.

Timothy R. Thomas received his B.A. in psychology from the University of California, Berkeley, in 1964 and his M.A. and Ph.D. from Tulane University in 1968 and 1969. He came to the Laboratory in 1986 as an Association of Western University Fellow and in 1989 joined the Computer User Services Group as a technical staff member. Thomas's current research efforts include implementing a system of delivering archival images of scientific papers to distributed users and developing computerized methods of extracting information from large corpora of text documents. His past research focused on neural networks as tools for mapping from speech to tongue movement. He has published research on perception, neural control of behavior, and hormonal physiology. Thomas has recently returned from Borneo where he assisted his daughter in observing orangutans for a Harvard University rain-forest-ecology project. He is currently preparing a 12-million-year-old rhinoceros skull that he found near Española, New Mexico, for display at Northern New Mexico Community College.

James M. White is section leader of the Computer Research and Applications Group and is responsible for designing and developing software applications in digital-processing techniques and applying these techniques to a wide range of scientific fields. He also initiates, executes, and manages new projects pertaining to image processing. White received his B.S. and M.S. in civil engineering from the University of Maryland in 1979 and 1982. Before joining the Laboratory in 1985, he was a Faculty Research Associate and Research Engineer at the University of Maryland and served as a Digital Imagery Scientist/Acting Chief of the Interactive Digital Image Manipulation Branch of the Central Intelligence Agency. White is a consultant in image and signal processing to government agencies, private firms, and universities. He was a member of the NASA Committee on Image Processing of the Shuttle Challenger Disaster in 1986. In 1988 White was issued a patent for a Monte Carlo vector quantizer.

Clustering and the Continuous k -Means Algorithm

Vance Faber

Many types of data analysis, such as the interpretation of Landsat images discussed in the accompanying article, involve datasets so large that their direct manipulation is impractical. Some method of data compression or consolidation must first be applied to reduce the size of the dataset without losing the essential character of the data. All consolidation methods sacrifice some detail; the most desirable methods are computationally efficient and yield results that are—at least for practical applications—representative of the original data. Here we introduce several widely used algorithms that consolidate data by clustering, or grouping, and then present a new method, the continuous k -means algorithm,* developed at the Laboratory specifically for clustering large datasets.

Clustering involves dividing a set of data points into non-overlapping groups, or clusters, of points, where points in a cluster are “more similar” to one another than to points in other clusters. The term “more similar,” when applied to clustered points, usually means closer by some measure of proximity. When a dataset is clustered, every point is assigned to some cluster, and every cluster can be characterized by a single reference point, usually an average of the points in the cluster. Any particular division of all points in a dataset into clusters is called a partitioning.

One of the most familiar applications of clustering is the classification of plants or animals into distinct groups or species. However, the main purpose of clustering Landsat data is to reduce the size and complexity of the dataset. Data reduction is accomplished by replacing the coordinates of each point in a cluster with the coordinates of that cluster’s reference point. Clustered data require considerably less storage space and can be manipulated more quickly than the original data. The value of a particular clustering method will depend on how closely the reference points represent the data as well as how fast the program runs.

A common example of clustering is the consolidation of a set of students’ test scores, expressed as percentages, into five clusters, one for each letter grade A, B, C, D, and F (see Figure 1). The test scores are the data points, and each cluster’s reference point is the average of the test scores in that cluster. The letter grades can be thought of as symbolic replacements for the numerical reference points.

Test scores are an example of one-dimensional data; each data point represents a single measured quantity. Multidimensional data can include any number of measurable attributes; a biologist might use four attributes of duck bills (four-dimensional data: size, straightness, thickness, and color) to sort a large set of ducks into several species. Each independent characteristic, or measurement, is one dimension. The consolidation of large, multidimensional datasets is the main pur-

* The continuous k -means algorithm is part of a patented application for improving both the processing speed and the appearance of color video displays. The application is commercially available for Macintosh computers under the names *Fast Eddie*, ©1992 and *Planet Color*, ©1993, by Paradigm Concepts, Inc., Santa Fe, NM. This software was developed by Vance Faber, Mark O. Mundt, Jeffrey S. Saltzman, and James M. White.

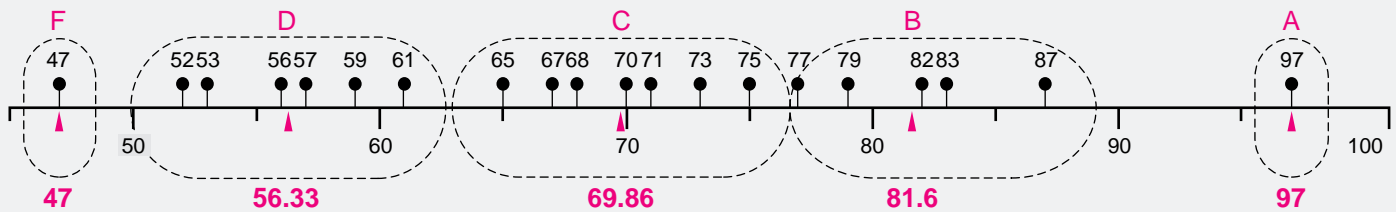


Figure 1. Clustering Test Scores
The figure illustrates an arbitrary partitioning of 20 test scores into 5 non-overlapping clusters (dashed lines), corresponding to 5 letter grades. The reference points (means) are indicated in red.

pose of the field of cluster analysis. We will describe several clustering methods below. In all of these methods the desired number of clusters k is specified beforehand. The reference point z_i for the cluster i is usually the centroid of the cluster. In the case of one-dimensional data, such as the test scores, the centroid is the arithmetic average of the values of the points in a cluster. For multi-dimensional data, where each data point has several components, the centroid will have the same number of components and each component will be the arithmetic average of the corresponding components of all the data points in the cluster.

Perhaps the simplest and oldest automated clustering method is to combine data points into clusters in a pairwise fashion until the points have been condensed into the desired number of clusters; this type of agglomerative algorithm is found in many off-the-shelf statistics packages. Figure 2 illustrates the method applied to the set of test scores given in Figure 1.

There are two major drawbacks to this algorithm. First—and absolutely prohibitive for the analysis of large datasets—the method is computationally inefficient. Each step of the procedure requires calculation of the distance between every possible pair of data points and comparison of all the distances. The second difficulty is connected to a more fundamental problem in cluster analysis: Although the algorithm will always produce the desired number of clusters, the centroids of these clusters may not be particularly representative of the data.

What determines a “good,” or representative, clustering? Consider a single cluster of points along with its centroid or mean. If the data points are tightly clustered around the centroid, the centroid will be representative of all the points in that cluster. The standard measure of the spread of a group of points about its mean is the variance, or the sum of the squares of the distance between each point and the mean. If the data points are close to the mean, the variance will be small. A generalization of the variance, in which the centroid is replaced by a reference point that may or may not be a centroid, is used in cluster analysis to indicate the overall quality of a partitioning; specifically, the error measure E is the sum of all the variances:

$$E = \sum_{i=1}^k \sum_{j=1}^{n_i} \|x_{ij} - z_i\|^2,$$

where x_{ij} is the j th point in the i th cluster, z_i is the reference point of the i th cluster, and n_i is the number of points in that cluster. The notation $\|x_{ij} - z_i\|$ stands for the distance between x_{ij} and z_i . Hence, the error measure E indicates the overall spread of data points about their reference points. To achieve a representative clustering, E should be as small as possible.

The error measure provides an objective method for comparing partitionings as well as a test for eliminating unsuitable partitionings. At present, finding the best

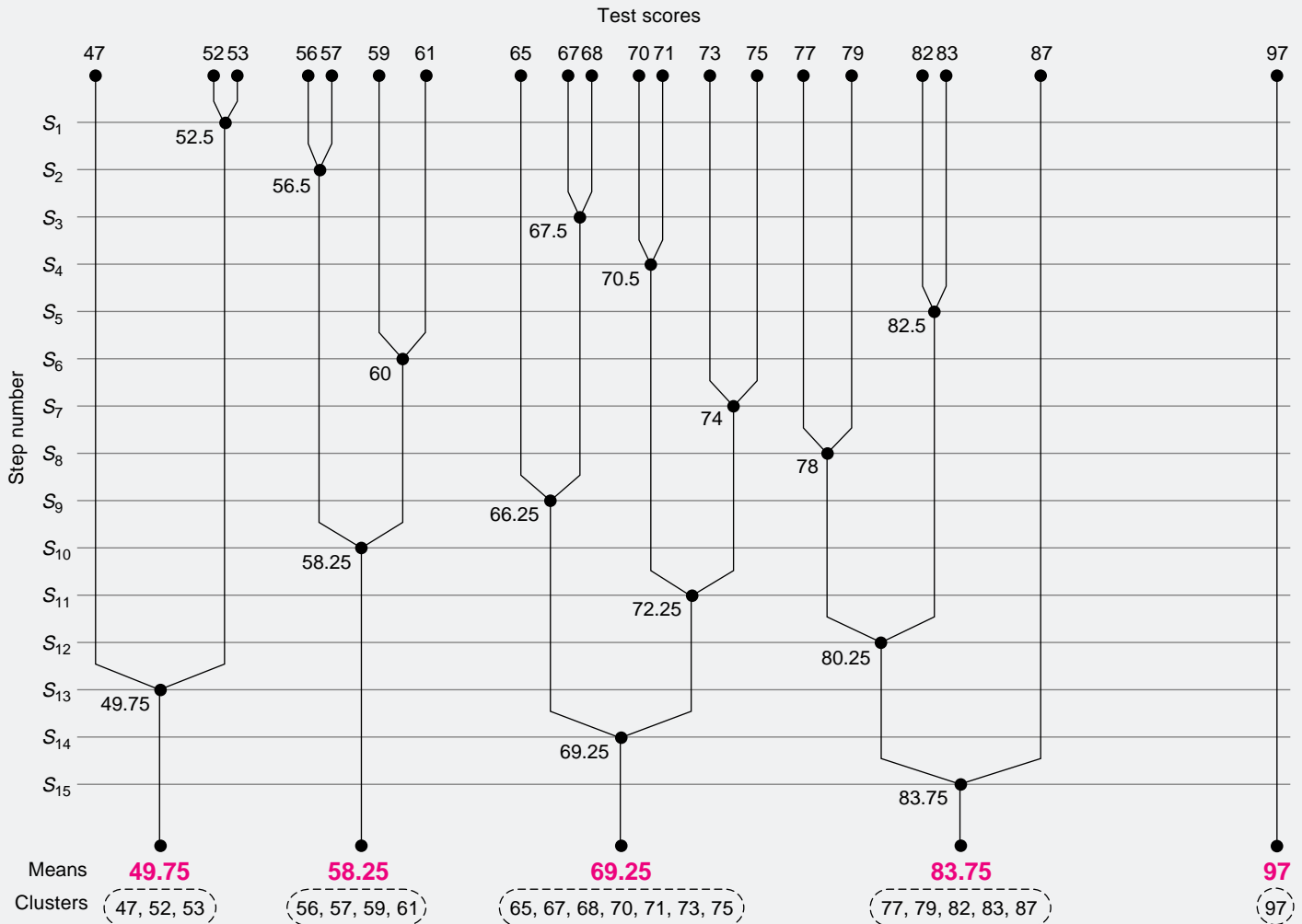


Figure 2. Pairwise Agglomerative Clustering

The figure illustrates the operation of an agglomerative clustering method, in which the 20 test scores of Figure 1 are successively merged by pairs of points and/or pairs of clusters until all the scores are collected into 5 clusters. The steps of the algorithm are shown in the branching of a dendrogram, or tree structure (much like a genealogy). A node, or branch point, indicates the merging of two branches into one, i.e. two data points into one cluster, or two clusters into one larger cluster. The algorithm begins with 20 separate clusters of one point apiece. For the first step in the algorithm, the closest two points (here, scores of 52 and 53) are found and merged into one cluster {52,53}. The two individual points are replaced by a single point equal to the unweighted average of the two points (52.5). The next step repeats this process (find the closest two points, calculate the average, merge the points), but with 19 points and 19 clusters (18 one-point clusters, plus 1 two-point cluster). There will be only one new branch, or merge at each step. Hence, if there is more than one pair of points at the minimum distance, only one pair will be merged at each step. It takes 15 steps to consolidate 20 points into 5 clusters.

partitioning (the clustering most representative of an arbitrary dataset) requires generating all possible combinations of clusters and comparing their error measures. This can be done for small datasets with a few dozen points, but not for large sets—the number of different ways to combine 1 million data points into 256 clusters, for example, is $256^{1,000,000}/256!$, where $256!$ is equal to $256 \times 255 \times 254 \times \cdots \times 2 \times 1$. This number is greater than $10^{2,000,000}$, or 1 followed by 2 million zeros.

When clustering is done for the purpose of data reduction, as in the case of the Landsat images, the goal is not to find the best partitioning. We merely want a reasonable consolidation of N data points into k clusters, and, if necessary, some efficient way to improve the quality of the initial partitioning. For that purpose, there is a family of iterative-partitioning algorithms that is far superior to the agglomerative algorithm described above.

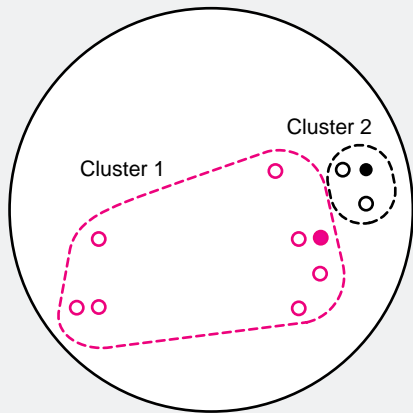
Iterative algorithms begin with a set of k reference points whose initial values are usually chosen by the user. First, the data points are partitioned into k clusters: A data point x becomes a member of cluster i if z_i is the reference point closest to x . The positions of the reference points and the assignment of the data points to clusters are then adjusted during successive iterations. Iterative algorithms are thus similar to fitting routines, which begin with an initial “guess” for each fitted parameter and then optimize their values. Algorithms within this family differ in the details of generating and adjusting the partitions. Three members of this family are discussed here: Lloyd’s algorithm, the standard k -means algorithm, and a continuous k -means algorithm first described in 1967 by J. MacQueen and recently developed for general use at Los Alamos.

Conceptually, Lloyd’s algorithm is the simplest. The initial partitioning is set up as described above: All the data points are partitioned into k clusters by assigning each point to the cluster of the closest reference point. Adjustments are made by calculating the centroid for each of those clusters and then using those centroids as reference points for the next partitioning of all the data points. It can be proved that a local minimum of the error measure E corresponds to a “centroidal Voronoi” configuration, where each data point is closer to the reference point of its cluster than to any other reference point, and each reference point is the centroid of its cluster. The purpose of the iteration is to move the partition closer to this configuration and thus to approach a local minimum for E .

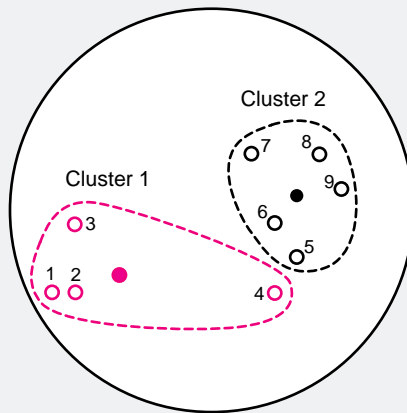
For Lloyd’s and other iterative algorithms, improvement of the partitioning and convergence of the error measure E to a local minimum is often quite fast—even when the initial reference points are badly chosen. However, unlike guesses for parameters in simple fitting routines, slightly different initial partitionings generally do not produce the same set of final clusters. A final partitioning will be better than the initial choice, but it will not necessarily be the best possible partitioning. For many applications, this is not a significant problem. For example, the differences between Landsat images made from the original data and those made from the clustered data are seldom visible even to trained analysts, so small differences in the clustered data are even less important. In such cases, the judgment of the analyst is the best guide as to whether a clustering method yields reasonable results.

(a) Setup:

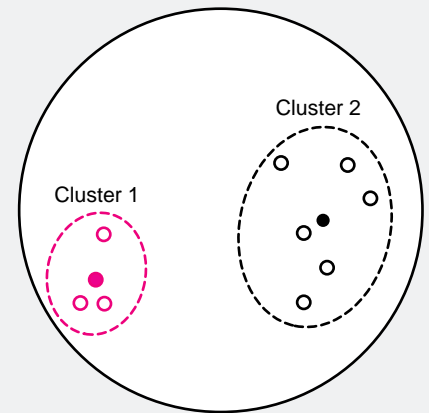
Reference point 1 (filled red circle) and reference point 2 (filled black circle) are chosen arbitrarily. All data points (open circles) are then partitioned into two clusters: each data point is assigned to cluster 1 or cluster 2, depending on whether the data point is closer to reference point 1 or 2, respectively.

**(b) Results of first iteration:**

Next each reference point is moved to the centroid of its cluster. Then each data point is considered in the sequence shown. If the reference point closest to the data point belongs to the other cluster, the data point is reassigned to that other cluster, and both cluster centroids are recomputed.

**(c) Results of second iteration:**

During the second iteration, the process in Figure 3(b) is performed again for every data point. The partition shown above is stable; it will not change for any further iteration.

**Figure 3. Clustering by the Standard k -Means Algorithm**

The diagrams show results during two iterations in the partitioning of nine two-dimensional data points into two well-separated clusters, using the standard k -means algorithm. Points in cluster 1 are shown in red, points in cluster 2 are shown in black; data points are denoted by open circles and reference points by filled circles. Clusters are indicated by dashed lines. Note that the iteration converges quickly to the correct clustering, even for this bad initial choice of the two reference points.

The standard k -means algorithm differs from Lloyd's in its more efficient use of information at every step. The setup for both algorithms is the same: Reference points are chosen and all the data points are assigned to clusters. As with Lloyd's, the k -means algorithm then uses the cluster centroids as reference points in subsequent partitionings—but the centroids are adjusted both during and after each partitioning. For data point x in cluster i , if the centroid z_i is the nearest reference point, no adjustments are made and the algorithm proceeds to the next data point. However, if the centroid z_j of the cluster j is the reference point closest to data point x , then x is reassigned to cluster j , the centroids of the “losing” cluster i (minus point x) and the “gaining” cluster j (plus point x) are recomputed, and the reference points z_i and z_j are moved to their new centroids. After each step, every one of the k reference points is a centroid, or mean, hence the name “ k -means.” An example of clustering using the standard k -mean algorithm is shown in Figure 3.

There are a number of variants of the k -means algorithm. In some versions, the error measure E is evaluated at each step, and a data point is reassigned to a different cluster only if that reassignment decreases E . In MacQueen's original paper on the k -means method, the centroid update (assign data point to cluster, recompute the centroid, move the reference point to the centroid) is applied at each step in the initial partitioning, as well as during the iterations. In all of these cases, the standard k -means algorithm requires about the same amount of computation for a single pass through all the data points, or one iteration, as does Lloyd's algorithm. However, the k -means algorithm, because it constantly updates the clusters, is unlikely to require as many iterations as the less efficient Lloyd's algorithm and is therefore considerably faster.

The Continuous k -Means Algorithm

The continuous k -means algorithm is faster than the standard version and thus extends the size of the datasets that can be clustered. It differs from the standard version in how the initial reference points are chosen and how data points are selected for the updating process.

In the standard algorithm the initial reference points are chosen more or less arbitrarily. In the continuous algorithm reference points are chosen as a random sample from the whole population of data points. If the sample is sufficiently large, the distribution of these initial reference points should reflect the distribution of points in the entire set. If the whole set of points is densest in Region 7, for example, then the sample should also be densest in Region 7. When this process is applied to Landsat data, it effectively puts more cluster centroids (and the best color resolution) where there are more data points.

Another difference between the standard and continuous k -means algorithms is the way the data points are treated. During each complete iteration, the standard algorithm examines all the data points in sequence. In contrast, the continuous algorithm examines only a random sample of data points. If the dataset is very large and the sample is representative of the dataset, the algorithm should converge much more quickly than an algorithm that examines every point in sequence. In fact, the continuous algorithm adopts MacQueen's method of updating the centroids during the initial partitioning, when the data points are first assigned to clusters. Convergence is usually fast enough so that a second pass through the data points is not needed.

From a theoretical perspective, random sampling represents a return to MacQueen's original concept of the algorithm as a method of clustering data over a continuous space. In his formulation, the error measure E_i for each region R_i is given by

$$E_i = \int_{x \in R_i} \rho(x) \|x - z_i\|^2 dx,$$

where $\rho(x)$ is the probability density function, a continuous function defined over the space, and the total error measure E is given by the sum of the E_i 's. In MacQueen's concept of the algorithm, a very large set of discrete data points can be thought of as a large sample—and thus a good estimate—of the continuous probability density $\rho(x)$. It then becomes apparent that a random sample of the dataset can also be a good estimate of $\rho(x)$. Such a sample yields a representative set of cluster centroids and a reasonable estimate of the error measure without using all the points in the original dataset.

These modifications to the standard algorithm greatly accelerate the clustering process. Since both the reference points and the data points for the updates are chosen by random sampling, more reference points will be found in the densest regions of the dataset and the reference points will be updated by data points in the most critical regions. In addition, the initial reference points are already members of the dataset and, as such, require fewer updates. Therefore, even when applied to a large dataset, the algorithm normally converges to a solution after only a small fraction (10 to 15 percent) of the total points have been examined. This rapid convergence distinguishes the continuous k -means from less efficient algorithms. Clustering with the continuous k -means algorithm is about ten times faster than clustering with Lloyd's algorithm.

The computer time can be further reduced by making the individual steps in the algorithm more efficient. A substantial fraction of the computation time required by any of these clustering algorithms is typically spent in finding the reference point closest to a particular data point. In a “brute-force” method, the distances from a given data point to all of the reference points must be calculated and compared. More elegant methods of “point location” avoid much of this time-consuming process by reducing the number of reference points that must be considered—but some computational time must be spent to create data structures. Such structures range from particular orderings of reference points, to “trees” in which reference points are organized into categories. A tree structure allows one to eliminate entire categories of reference points from the distance calculations. The continuous k -means algorithm uses a tree method to cluster three-dimensional data, such as pixel colors on a video screen. When applied to seven-dimensional Landsat data, the algorithm uses single-axis boundarizing, which orders the reference points along the direction of maximum variation. In either method only a few points need be considered when calculating and comparing distances. The choice of a particular method will depend on the number of dimensions of the dataset.

Two features of the continuous k -means algorithm—convergence to a feasible group of reference points after very few updates and greatly reduced computer time per update—are highly desirable for any clustering algorithm. In fact, such features are crucial for consolidating and analyzing very large datasets such as those discussed in the accompanying article. □

Further Reading

James M. White, Vance Faber, and Jeffrey S. Saltzman. 1992. Digital color representation. U.S. Patent Number 5,130,701.

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* IT-28: 129–137.

Edward Forgy. 1965. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications, *Biometrics* 21: 768.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I, Statistics*. Edited by Lucien M. Le Cam and Jerzy Neyman. University of California Press.

Jerome H. Friedman, Forest Baskett, and Leonard J. Shustek. 1975. An algorithm for finding nearest neighbors. *IEEE Transactions on Computers* C-24: 1000-1006. [Single-axis boundarizing, dimensionality.]

Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3: 209–226. [Tree methods.]

Helmuth Späth. 1980. *Cluster Analysis Algorithms for Data Reduction and Classification of Objects*. Halsted Press.

Anil K. Jain and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall.

The biography of Vance Faber appears on page 149.

Digital Village

John D. MacCuish, Susan M. Mniszewski, Gregory E. Shannon, and Bonnie C. Yantis



The Digital Village Initiative is a telecommunity-outreach project of the Laboratory's Computer Research and Applications Group. A

telecommunity is a society that exists and functions through electronic communication. There are two types of telecommunities. The first is quite in-

dependent of geography—global, independent telecommunities exist for everything from auto repair to environmental activism to molecular biology.

The graphic at left and those scattered throughout this article are reproduced from a user-interface prototype for telecommunities that was developed at the Laboratory.

The second has its roots in a specific geographical location and is predicated on the notion that people who live in a particular town, or county, or region naturally share many interests. Established communities already have organized social and political structures, such as local government, civic groups, and business alliances, but the introduction of an electronic-telecommunity structure can facilitate and broaden many aspects of their interactions.

Several technological, economic, and social factors were influential in bringing about the emergence of telecommunities. Over the last thirty years computer hardware has become increasingly more powerful and less expensive, and the personal computer is now ubiquitous. Technologies and standards for connecting computer networks have become more efficient and robust. The Internet, the computer network used by most telecommunities, has its roots in the U.S. military's desire for an electronic-communication infrastructure that would survive a nuclear attack—the system as a whole had to remain operational even if portions of it were destroyed. In 1968, the Advanced Research Projects Agency (ARPA) initiated a prototype system, ARPAnet, with that design requirement in mind.

ARPAnet evolved as a research and development tool used by military and academic research centers around the country. In the mid-1980s the National Science Foundation (NSF) assumed much of the responsibility for administration of ARPAnet. The NSF then started its own network, NSFnet, for academic and commercial concerns, and also linked together other networks to form the NSFnet “backbone.” The result was a matrix of powerful computers with high-speed connections linking the various networks together. The NSF eliminated many of the restrictions set forth by ARPAnet's appro-



propriate-use policy, and the NSF network, or the Internet as we know it today, became an international research- and education-oriented network of over a thousand government, academic and commercial entities.

Once the strict usage policies were lifted, distinct cultures began appearing on the Internet to address a vast array of interactive human-communication needs (see Figure 1). Electronic mail—now almost as essential as the telephone for electronic communication—is the spontaneous outgrowth of technology originally developed for transmitting



scientific data. People routinely use e-mail for all types of communication from coordinating community calendars to writing their congressional representatives.

The Digital Village Initiative was launched to design and test infrastructure technologies for both local and na-

tional telecommunities. The outreach initiative operates through collaborations between Laboratory personnel and members of proposed or emerging telecommunities. That approach ensures that the technical choices associated with telecommunity development meet the specific needs of each community within the context of its culture. The collaborative work was motivated by contact with a number of emerging New Mexico telecommunities including those of San Ildefonso Pueblo, Taos, Santa Fe, Las Vegas, Farmington, and Los Alamos. In addition, the project is a response to the current administration's initiatives to develop and implement the technology for the National Information Infrastructure (NII), and it complements the strategic shift of both the Department of Energy (DOE) and the Laboratory toward greater emphasis on the nation's economic security.

Rapid advances in hardware and software present emerging telecommunities with options that are often beyond their resources to evaluate or implement. The Laboratory's strong capabilities as a premier computing, information, and communications research center are a valuable resource for evaluating available technology and recommending equipment and systems to meet the needs of a variety of telecommunities. Our abilities to test equipment in various configurations, to model customer applications, to evaluate human-computer interactions, and to analyze the performance of prototype systems will be a significant force in determining the overall scope and character of the NII.

Among the many applications being explored and modeled as part of the Digital Village Initiative are:

- * telemedicine, through which doctors can consult with one another more effectively by sending

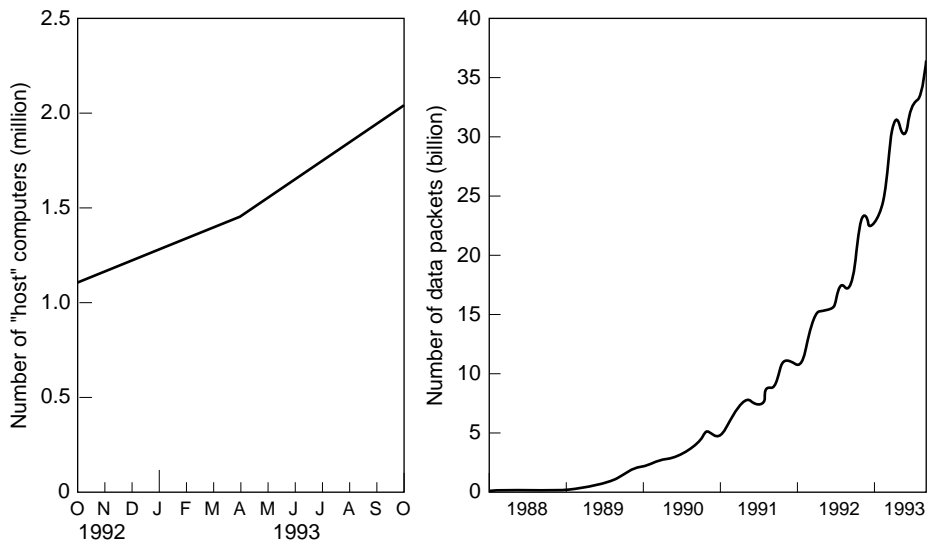


Figure 1. Growth of the Internet

The graph on the left shows, for a twelve-month period beginning October 1992, the number of computers (in millions) acting as “gateways” to the Internet, a system of networks once shared by only a few thousand users. The graph on the right shows the dramatic increase in network traffic over the last five calendar years. Plotted in the graph are the number of data packets (each equivalent to 200 typewritten characters) transmitted each month over the backbone of the Internet.

medical records or x-ray images, for example, over the Internet;

- * electronic commerce through which almost any type of business can be conducted over a network such as CommerceNet;
- * digital libraries, which will make local, national, and international libraries available to all members of the telecommunity;
- * telecommuting, through which more and more people will be able to work from their homes;
- * electronic classrooms, which will free students and teachers from the constraints of time and location;
- * and electronic government services, which will provide information and facilitate many different types of transactions on a local, state, and federal level.

The collaborative nature of the Digital Village Initiative is in harmony with the underlying philosophy that telecommunities should be created and main-

tained by the people who “live” in them. The members of the telecommunities must decide themselves how they will provide access, whether they will guarantee universal access, how they will define appropriate use and content, and how they will fund their “mile on the information highway.”

Technical Issues

The Digital Village Initiative employs a wide spectrum of dual-use technologies originally developed through the Laboratory’s defense and nonproliferation efforts—specifically, research in the areas of computer security and privacy, high-performance computing, human-computer interaction, information mining, and network navigation. Our collaborations with emerging telecommunities are yielding many opportunities to assess interactions among technologies, applications, and policies as well as occasions to evaluate Laboratory research within new contexts. Technology transfer of software and hardware, for example, is central to the Initiative. A good prospect for transfer is software being developed by the

DOE Digital superLab. The goal of the superLab project is to facilitate collaboration among three National Defense Laboratories (Livermore National Laboratory, Los Alamos National Laboratory, and Sandia National Laboratories) through the use of advanced computational and information-management technology. Much of the software being developed for superLab is object-oriented; that is, it makes use of a set of modules, or building blocks, that can be assembled in various configurations appropriate for specific applications. Thus it can be reconfigured to meet the specific needs of a particular telecommunity.

Security and privacy. Security and privacy issues are fundamental to the successful implementation of telecommunities. Security is critical for electronic commerce and digital cash exchanges. Furthermore, if medical records or other sensitive information will be accessible, privacy must be guaranteed. For example, a financial or medical system must guarantee that its infrastructure and content will not be damaged, that no information is divulged during transmission, and that the integrity of the information can be verified. Wireless telecommunication technologies amplify security and privacy concerns because such transmissions are accessible to virtually anyone who cares to “listen.”

The Laboratory has a long history of contributions to computer-security technology. Our ability to evaluate and implement state-of-the-art encryption codes and intrusion-detection methods is being applied to the problem of safeguarding information transmitted across the network. We are currently working with the Internal Revenue Service and the Social Security Administration to explore a variety of access-control and user-identification techniques. In addi-

tion, we are working with the Financial Services Technical Consortium (FSTC) to examine the security issues related to electronic commerce. FSTC is a coalition of large banks, the Department of the Treasury, the Department of Commerce, and the DOE national laboratories. The coalition's mission is to facilitate nation-wide electronic commerce.

High-performance computing.

High-performance computing is fundamental to the NII and its support of telecommunities. The sheer volume and complexity of data that will be made available over the Internet necessitates high-bandwidth communications and high-performance-computing resources. Consider the staggering quantity of related but quite heterogeneous information that would be necessary to provide a U.S. geographic-information database, the current conditions for all modes of transportation around the country, and the current tariffs for all common carriers. Manufacturers, shippers, and service agencies would benefit tremendously from quick, easy access to information through which they



could determine the best mode of transport for their purposes. Such a system would require high-performance-computing resources including high-speed

processing and a great deal of memory. Our expertise in the parallel utilization, scheduling, and allocation of distributed computing resources will help make services of that kind a reality.

Human-computer interfaces. Creating effective human-computer interfaces for the NII or for any particular telecommunity will be challenging because of the diversity within and among communities. To make telecommunities accessible to the greatest possible number of people, we are implementing an icon-based user interface with multimedia capabilities. That interface is also being developed in conjunction with the superLab Project. The use of simple pictures or symbols will accommodate people who are unfamiliar or uncomfortable with computers. Other aspects of the interface might accommodate users who have various disabilities or who do not read or write English. For instance the interface might include audio capabilities or Braille labeling to assist sight-impaired users. Interfaces that respond to voice commands and furnish audio feedback could address a variety of user-interface issues. To build audio-driven systems, however, we must deal with the technical challenge of incorporating voice input/output into existing applications. As we plan prototype computer interfaces, we can take advantage of modern design tools that allow us to iteratively present scenarios to potential users and incorporate their evolving requirements into the design.

Information mining and network navigation. The quantity of electronic information currently available through the Internet is overwhelming. Moreover, the information exists in a wide variety of formats, including video, sound, and text. It is not uniformly accessed, organized, or stored nor is all

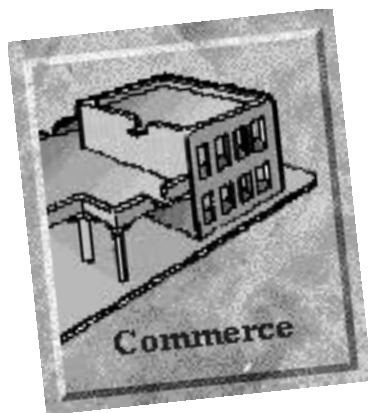


the information about a specific topic located in one place. The entire nation faces the challenge of developing an infrastructure that will allow efficient access to and use of information. The Laboratory is a leader in the management, visualization, and analysis of extremely large and complex datasets. The Laboratory's IBROWSE project allows doctors to view CT lung-scan images and search large databases for similar lung-scan images. Other supporting technologies include an innovative data-compression algorithm based on a mathematical technique called wavelet transforms. The Federal Bureau of Investigation has adopted this technique as a national standard for compressing digitized fingerprint records.

Several Laboratory research projects involve the development of tools that will allow a user to put a query out on the Internet and receive replies from a variety of sources. The results of the query, "Tell me about the University of New Mexico," for example, might yield a video "tour" of the campus, a sound-clip from a radio interview with the president of the University, an undergraduate catalog, a tuition spreadsheet, and an accreditation report—and each might come from a different computer system. Efficient information-mining techniques require that all of the articles, newsgroups, databases, and so forth that are accessible through the

network have some form of quantitative descriptor, perhaps obtained by an analysis of patterns of characters (see “Concept Extraction—A Data-Mining Technique”). Statistical clustering techniques could then be used to sort through the entire available dataset and obtain only the desired information.

Network navigation tools, which will help users “move” through the Internet, are being developed as part of the Laboratory’s Sunrise Project. One such navigation tool presents a visual representation of the “path” a user is taking through the Internet and also initiates Internet searches by communicating with WorldWideWeb servers and Mosaic (an Internet information-browser interface). That tool allows users to search and navigate the Internet more easily and to document the paths of their searches for future reference.



Kiosks and universal access. The concept of installing kiosks to provide information and services was first realized in the early 1980s with the advent of the automated teller machine, or ATM. More recently a variety of government agencies have become interested in installing kiosks as a means of cutting costs while making services more readily available. In support of

Vice President Gore’s National Performance Review, the Laboratory is collaborating with a wide variety of federal agencies to develop systems that provide access to government information. Local telecommunities might choose to collaborate with these agencies and incorporate government services into a network of kiosks in locations suited to the needs of the community. A kiosk-based telecommunity is the most feasible means of guaranteeing universal access to the information superhighway at a modest cost. Of course, any personal computer will be an “information portal” to the National Information Infrastructure. However, if local telecommunities install kiosks in public places, the kiosks will serve as information portals for those who do not have access to personal computers.

Telecommunity Applications

A wide range of applications are already in use in a limited form and are only awaiting advances in technology before their full potential can be realized. For example, we are currently working with the Indian Health Service (IHS) to establish a wireless “healthnet” among Native American communities in New Mexico. The network will connect mobile units with regional clinics and the medical center at the University of New Mexico. To bring additional medical expertise to this isolated, rural health network, we are also teaming with National Jewish Hospital in Denver. National Jewish is the recognized leader in pulmonary-disease research. Drug-resistant tuberculosis occurs frequently in reservation populations, and we believe that the collaboration between the IHS and National Jewish Hospital will have immediate positive effects. The IHS healthnet can serve as a prototype for a national healthnet.

Undoubtedly, local telecommunities will provide some form of access to public-sector services and private-sector commerce. Students living in rural communities will have the opportunity to take courses and “attend” lectures at universities around the country. “Electronic visits” to the local Department of Motor Vehicles to renew registrations or transfer titles will reduce administrative costs—and there will be no need to stand in line! Telecommunities are a natural conduit to a dynamic and competitive global marketplace where commerce flows freely among local, national, and international communities. “Electronic showrooms” connected to the Internet will be perpetually open to world markets. With a greatly expanded customer base but no need for the traditional showroom expenses, small businesses will have more opportunities for growth.

The development of individual, local telecommunities as a series of pilot projects offers an innovative approach to designing the nation’s Information Superhighway. Communities can determine their own needs, make their own choices, and ultimately serve as a market force that will shape development at the national level. New Mexico in particular provides a rich testing-ground for the development of telecommunities. Here in New Mexico there are both rural and urban areas, there are networks of small communities separated by considerable distances, and there are several distinct cultures throughout the state.

The Clinton administration’s goal for the NII is that all people have access to the Information Superhighway. As the technology advances, there is growing concern that the design and structure of the coming web of communication technologies will be driven primarily by commercial interests and thus exclude large sectors of the population through

economic and educational barriers. The 1993 National Telecommunications and Information Administration (NTIA) report concludes that:

An advanced information infrastructure will enable U.S. firms to compete and win in the global economy, generating good jobs for the American people and economic growth for the nation. As importantly, the NII can transform the lives of the American people—ameliorating the constraints of geography, disability, and economic status—giving all Americans a fair opportunity to go as far as their talents and ambitions will take them.

For such a vision to be realized, we must strike a balance between commercial interests and the government's goal of universal public access. Through the Digital Village Initiative the Laboratory is helping the nation realize this vision by facilitating the development of local telecommunities in which broad-based community agendas can flourish. ■

Acknowledgements

Contributors to the Initiative include: Stephen J. Adelson, Linda K. Anderson, Christopher Barnes, Nathaniel M. Bobbitt, William E. Bostwick, Aaron C. Coday, James M. Cruz, John R. Deal, Vance Faber, Patricia K. Fasel, David W. Forslund, Anne T. Gilman, Diane M. Gonzales, Judith G. Hochberg, John A. Hopkins, Kathleen A. Jackson, Patrick M. Kelly, Robert L. Kelsey, David G. Kilman, Kevin S. Klenk, Emanuel H. Knill, Patrick S. McCormick, James P. McGee, Allen L. McPhearson, Donald R. Montoya, Gerald D. Morris, Ronald T. Pfaff, Richard L. Phillips, Reid D. Rivenburgh, Torrin D. Sanders, Timothy R. Thomas, Richard A. Ulbarrim, and Arthur L. Wilson.

We would like to thank our management and the people supporting the initiative: William L. Thompson, Andrew B. White Jr., Hassan A. Dayem, Hans M. Ruppel. We would also like to thank Bill Enloe of Los Alamos National Bank and Jane Proudy and Marcia Martinez from the State of New Mexico.



Gregory E. Shannon, Susan M. Mniszewski, Bonnie C. Yantis, and John D. MacCuish

Further Reading

Brendon P. Kehoe. 1994. *Zen and the Art of the Internet: A Beginners Guide*. Prentice Hall.

Ed Krol. 1994. *The Whole Internet: User's Guide & Catalog*. O'Reilly & Associates, Inc.

Hans Klein and Coralee Whitcomb. 1994. Developing an equitable and open information infrastructure. In *Proceedings of A Directions and Implications of Advanced Computing Symposium*. Computer Professionals for Social Responsibility.

Office of President Bill Clinton. 1993. *Technology for Economic Growth: President's Progress Report*. U.S. Government Printing Office.

U.S. Department of Housing and Urban Development, Office of Community Planning and Development. *Building Communities: Together. Guidebook for Community-Based Strategic Planning for Empowerment Zones and Enterprise Communities*. U.S. Government Printing Office. HUD-1442-CPD.

U.S. Department of Agriculture, Office of Small Community and Rural Development. 1994. *Building Communities: Together. Empowerment Zones and Enterprise Communities Application Guide*. U.S. Government Printing Office. HUD-1445-CPD.

Office of Science and Technology Policy. 1994. *High Performance Computing and Communications: Toward a National Information Infrastructure*. U.S. Government Printing Office.

John D. MacCuish received his B.A. in philosophy and fine art from the University of California, Santa Barbara, in 1978, and his M.S. in computer science from Indiana University, Bloomington in 1993. He is currently enrolled in the computer science doctoral program at the University of New Mexico, Albuquerque. MacCuish joined the Laboratory in 1993 as a graduate research assistant in the Computer Research and Applications Group.

Susan M. Mniszewski joined the Laboratory in 1978 as a staff member in the Computing Research and Applications Division. Her research interests include distributed computing, object oriented systems, and network simulation and modeling. Mniszewski earned her B.S. in computer science from the Illinois Institute of Technology in 1976. She received a patent in 1988 for encryption and transmission of digital keying data.

Gregory E. Shannon received his B.S. in computer science from Iowa State University in 1982, and his Ph.D. in computer sciences from Purdue University in 1988. He joined the Laboratory's Computer Research and Application Group in 1993. His research interests include empirical analysis of discrete algorithms, problem-solving environments, and information mining.

Bonnie C. Yantis joined the Laboratory in 1986 to work on functional language research. She has been deputy group leader for the Computer Research and Applications Group since 1991. She received her M.S. in computer science from the University of Nevada, Las Vegas. Yantis has been the general chair for the Computational Science Workshop for the past three years.



@xxx.lanl.gov

first steps toward electronic research communication

Paul H. Ginsparg



hep-th@xxx.lanl.gov is the e-mail address for the first of a series of automated archives for electronic communication of research information. This “e-print archive” went on-line in August, 1991. It began as an experimental means of circumventing recognized inadequacies of research journals, but unexpectedly became within a very

short period the primary means of communicating ongoing research information in formal areas of high energy particle theory. Its rapid acceptance within this community depended critically on both recent technological advances and particular behavioral aspects of the community. There are now more than 3600 regular users of hep-th worldwide.

The archiving software has been expanded to serve a number of other research disciplines (see Figure 1). The extended, automatically maintained database and distribution system currently serves over 20,000 users from more than 60 countries and processes over 30,000 messages per day. It is already one of the largest and most active

databases on the internet. This system may be a paradigm for worldwide, discipline-wide scientific-information exchange when the next generation of “electronic-data highways” begins to provide more universal access to high-speed computer networks.

Background

The rapid acceptance of electronic communication of research information in my own community of high-energy theoretical physics was facilitated by a pre-existing “preprint culture,” in which the irrelevance of refereed journals to ongoing research has long been recognized. Since the mid-1970s the primary means of communicating new research ideas and results has been a preprint-distribution system in which printed copies of papers were sent through the ordinary mail to large distribution lists at the same time that they were submitted to journals for publication. (The larger high-energy physics groups typically spent between \$15,000 and \$20,000 per year on copying, postage, and labor costs for their preprint distribution.) Typically, it takes six months to a year for a paper to appear in a journal. Members of our community have therefore learned to determine from the title and abstract (and occasionally the authors) whether we wish to read a paper as well as to verify results ourselves rather than rely on the alleged verification of overworked or otherwise careless referees. The small amount of filtering provided by refereed journals plays no effective role in our research.

Taking advantage of advances in computer software and hardware, many of us had already begun using highly informal mechanisms of electronic-information exchange by the mid-1980s. The first such advance was a program called TeX, which was written by com-

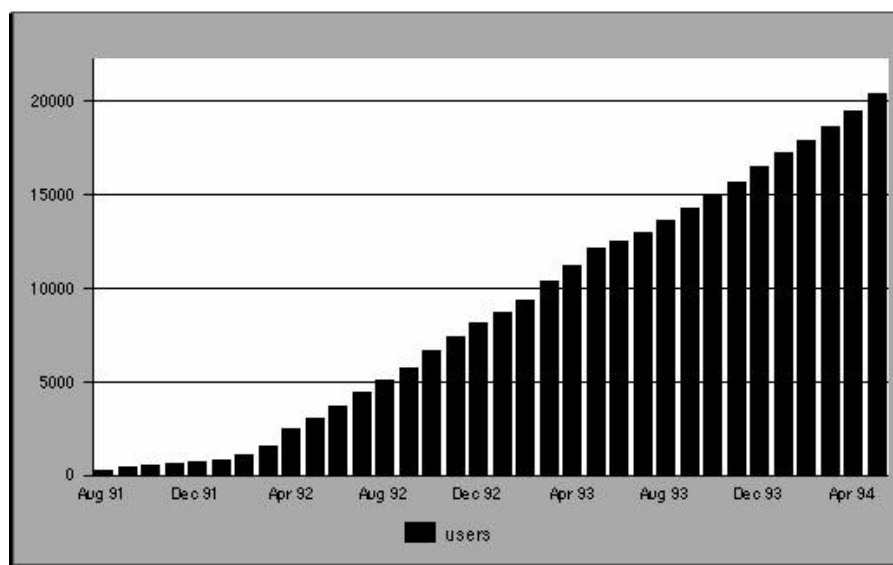


Figure 1. Number of Users of E-Print Archives

The bar chart shows the combined number of e-print-archive users over the period beginning August 1991 and ending April 1994. The data include users of the following e-print archives: High-energy particle theory (formal), started August 1991; Algebraic geometry, started February 1992; High-energy particle theory (phenomenological), started March 1992; Astrophysics, started April 1992; Condensed-matter theory, started April 1992; Computational and lattice physics, started April 1992; Functional analysis, started April 1992; General relativity/Quantum cosmology, started July 1992; Nuclear theory, started October 1992; Nonlinear Sciences, started March 1993; Economics, started July 1993; High-energy experimental physics, started April 1994; Chemical physics, started April 1994; Computation and language, started April 1994.

puter scientist Donald E. Knuth of Stanford. TeX was soon adopted as our standard scientific wordprocessor, and for the first time we could produce for ourselves a printed version equal or superior in quality to the published version. TeX has the additional virtue of being based on ASCII, so transmitting TeX files between different computer systems is straightforward. Collaboration at a distance became extraordinarily efficient, since we no longer had to express-mail versions of a paper back and forth and could instead see one another’s revisions essentially in real time. Figures and technical illustrations can also be generated within a TeX-oriented picture environment or, more generally, can be transmitted as stan-

dardized Postscript files produced by a variety of graphics programs.

A second technological advance achieved during the same period was the exponential increase in computer network connectivity. By the end of the 1980s, virtually all researchers in this community were plugged into one or another of the interconnected worldwide networks and were using e-mail on a daily basis. Finally, the development of large on-line archives of research papers has been enabled by the widespread availability of low-cost, high-powered workstations with high-capacity storage media. After compression, storing an average paper with figures requires 40 kilobytes of memory. Thus one of the current generation of

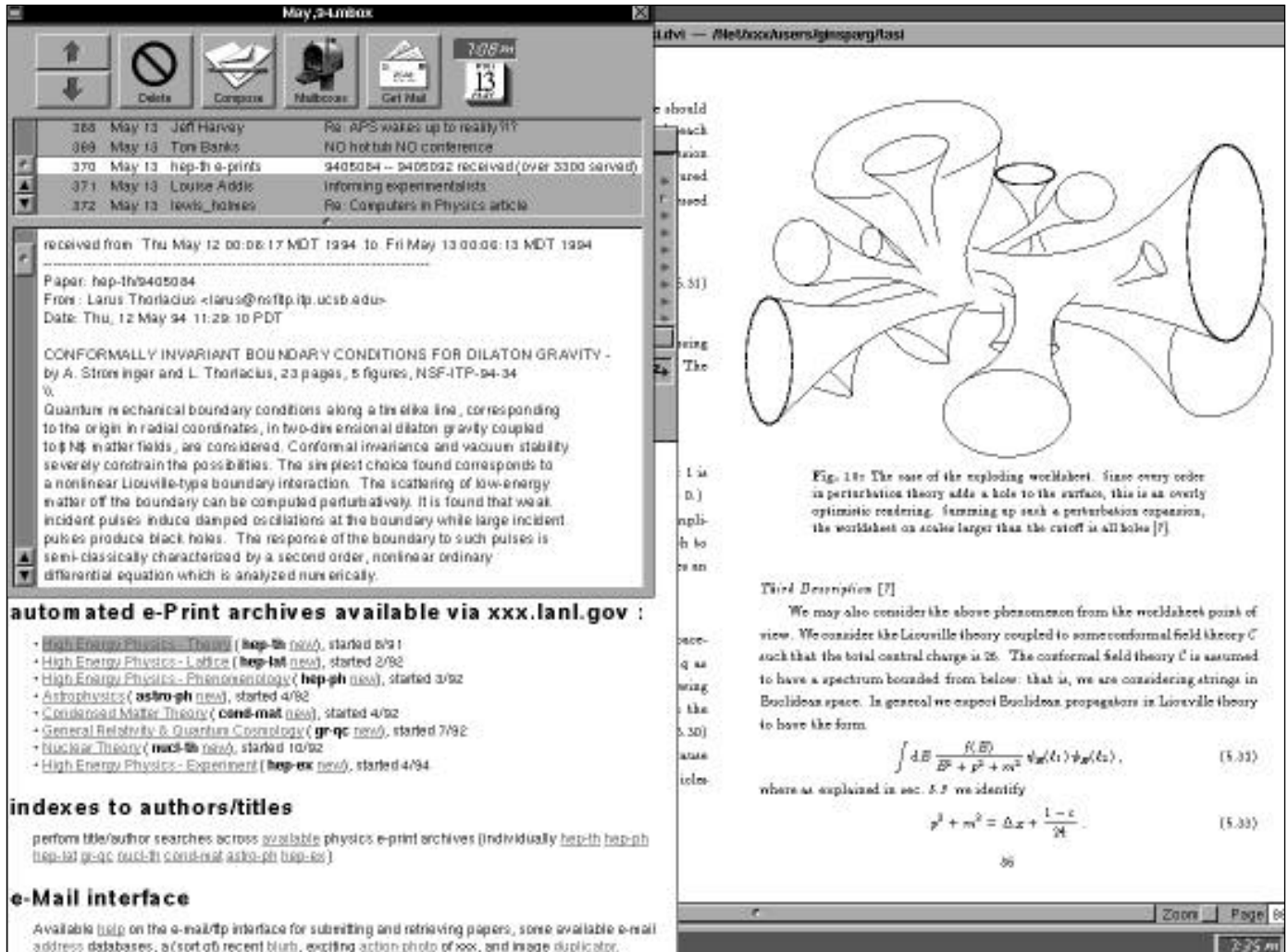


Figure 2. User Interface for E-Print Archives

The left side of the sample screen above shows two user interfaces for accessing the e-print archives. The window in the upper left corner shows abstracts received through e-mail, and the window in the lower left corner shows the graphical user interface provided by a WorldWideWeb client (in this case OmniWeb.app running under NeXTstep) accessing the frontpage <http://xxx.lanl.gov/>. (The underlined text signifies network hyperlinks that bring up new hypertext when clicked upon.) A paper extracted from the e-print archive appears in the window on the right side, where it can be read or sent to a printer.

rapid-access gigabyte disk drives costing under \$1,000 can hold 25,000 papers at an average cost of 4 cents per paper. Slower-access media for archival storage cost even less: A digital audio-tape cartridge, available from discount electronics dealers for under \$15, can hold over 4 gigabytes, that is, over 100,000 such papers. The data equivalent of multiple years of most journals is often far less than the amount many experimentalists handle every day. Moreover, the costs of data storage will only continue to decrease.

Since storage is so inexpensive, an archive can be duplicated at several distribution points, minimizing the risk of loss due to accident or catastrophe and facilitating worldwide network access. The Internet runs 24 hours a day—with virtually no interruptions—and transfers data at rates of up to 45 megabits per second (that is, less than a hundredth of a second per paper). Projected upgrades of NSFnet to a few gigabits per second within a few years should be adequate to accommodate increased usage for the academic community. The commercial networks that will constitute the nation's electronic data highway will have even greater capacity.

These technological advances—combined with a remarkable lack of response to the electronic revolution from conventional journals—rendered the development of e-print archives “an accident waiting to happen.” Perhaps more surprising has been the readiness of scientific communities to adopt this new tool of information exchange and to explore its implications for traditional review and publication processes. The exponential growth in archive usage suggests that scientific researchers are not only eager—but indeed impatient—for completion of the proposed “information superhighways” (though not necessarily the tollbooths of “information turnpikes”).

Implementation

Having concluded that an electronic preprint archive was possible in principle, I spent a few afternoons during the summer of 1991 writing the original software. It was designed as a fully automated system in which users construct, maintain, and revise a comprehensive database and distribution network without outside supervision or intervention. The software is rudimentary and allows users with minimal computer literacy to communicate e-mail requests to the Internet address `hep-th@xxx.lanl.gov`. Remote users can submit and replace papers, obtain papers and listings, get help on available commands, search the listings for author names, and so on.

The formal communication provided by an “e-print archive” should be distinguished from the informal (and unarchived) communication provided by electronic bulletin boards and network news. In the case of an e-print archive, researchers are restricted to communication by means of abstracts and research papers suitable for publication in conventional research journals. Electronic bulletin boards are more akin to ordinary conversation or written correspondence; that is, they are neither indexed for retrieval nor stored indefinitely. The e-print archives allow a submitter to replace his or her submission, and the program automatically checks on database integrity to ensure, for example, that the person replacing a submission is indeed the original submitter. In addition, the system maintains permanent records of submissions and the dates they were submitted, and it records the number of user requests for each paper. Subscribers to the system receive a daily listing of new titles and abstracts (see Figure 2).

The initial user base for `hep-th` was 160 addresses assembled from pre-ex-

isting e-mail distribution lists in the subject of two-dimensional gravity and conformal field theory. Within six months the user base grew to encompass most of the workers in formal quantum field theory and string theory, and now includes the 3600 subscribers mentioned above. Its smooth operation has transformed it into an essential research tool—many users have reported their dependence on receiving multiple “fixes” each day. The original `hep-th` archive now receives roughly 200 new submissions per month, responds to more than 700 e-mail requests per day, and transmits more than 1000 copies of papers on peak days. Internet e-mail access time is typically a few seconds. The system originally ran as a background job on a small UNIX workstation (a 25-megahertz NeXTstation with a 68040 processor purchased for roughly \$5,000 in 1991), which was primarily used for other purposes by another member of my research group, and placed no noticeable drain on CPU resources. The system has since been moved to an HP 9000/735 that sits exiled on the floor under a table in a corner.

For those directly on the Internet, the system allows anonymous FTP access to the papers and listings directories. Now access can be gained through WorldWideWeb for those with the required (public-domain) client software (see Figure 3). Local menu-driven interfaces can be set up to automatically pipe selected papers through text formatters directly to a screen previewer or printer. (Such software has been set up to cache and redistribute papers on many local networks.) The WorldWideWeb interface for the multiple archives at `xxx.lanl.gov` currently processes over 5000 requests daily (see Figure 4). While that is only a small fraction of the overall usage of the e-print archives, WorldWideWeb is ex-

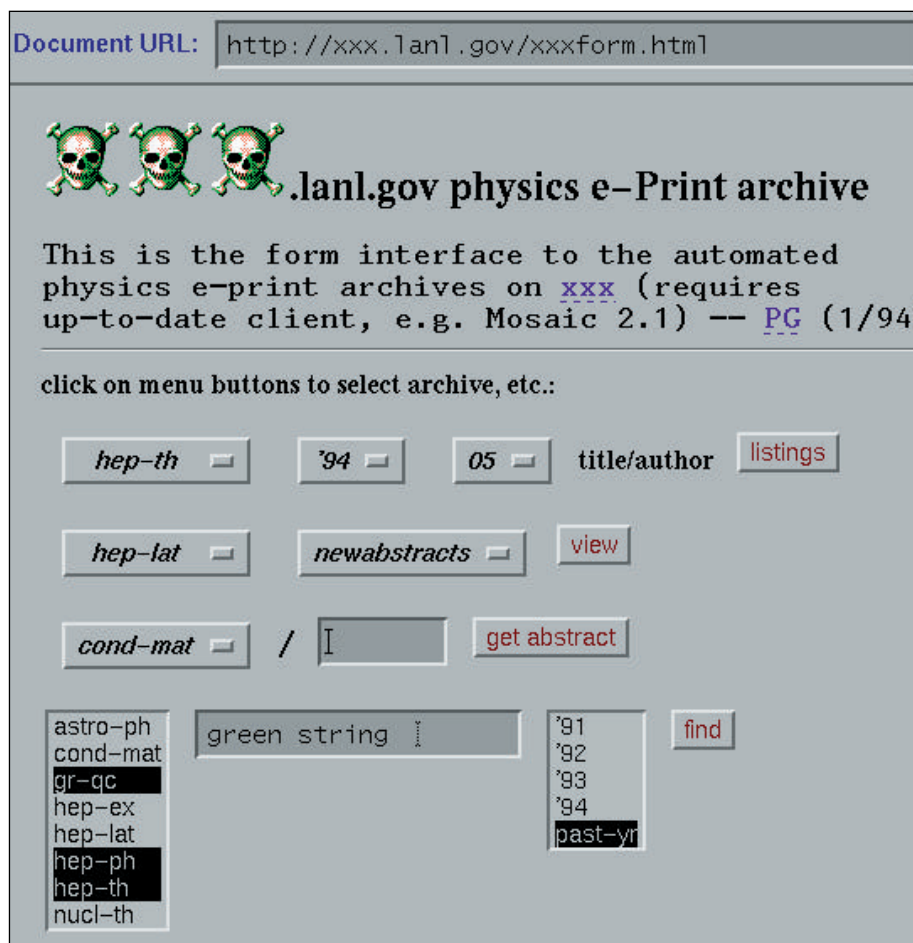


Figure 3. The “Form Interface” on xxx.lanl.gov

This screen grab shows another WorldWideWeb client, Mosaic, accessing the “form interface” on xxx.lanl.gov. The Motif “buttons” allow the client to choose an archive to view monthly listings or daily abstracts received, or to search the title/author listings of selected archives for given time periods. Listings are displayed in hypertext with included hyperlinks that retrieve paper abstracts or full text in either TeX or Postscript format.

pected to become the dominant mode of access.

An active archive such as hep-th requires about 70 megabytes per year (that is, \$70 per year) for storing papers, including figures. Its network usage is less than 10^{-4} of the lanl.gov backbone capacity, so it places a negligible drain on local network resources. It requires little intervention and has run entirely unattended for extended periods while I have been away on travel. It is difficult to estimate the potential of future dedicated systems because the resources of the current experimental system (run free of charge) are so far from saturation.

Storage and retrieval of figures. Although software for technical illustrations has not yet been standardized, the vast majority of networked physics institutions have screen previewers and laser-printers that display and print Postscript files created by a wide variety of graphics programs. Figure files are typically submitted as compressed Postscript, and papers can be printed with the figures embedded directly in the text. High-resolution digital scanners will soon become as commonplace as fax machines, thus permitting the inclusion of figures of almost any origin. (Of course it is already possible to fax figures in any format to a machine equipped with a fax modem, convert them to bitmapped Postscript files, and then append them to the paper.) Using appropriate data compression and Postscript conversion, figures typically increase paper-storage requirements by an inconsequential factor of 2.

Some measure of the success of e-print archives is given, first, by numerous testimonials from users that they find it an indispensable research tool—effectively eliminating their reliance on conventional print journals; second, by decisions of numerous institutions to discontinue their preprint mailings in

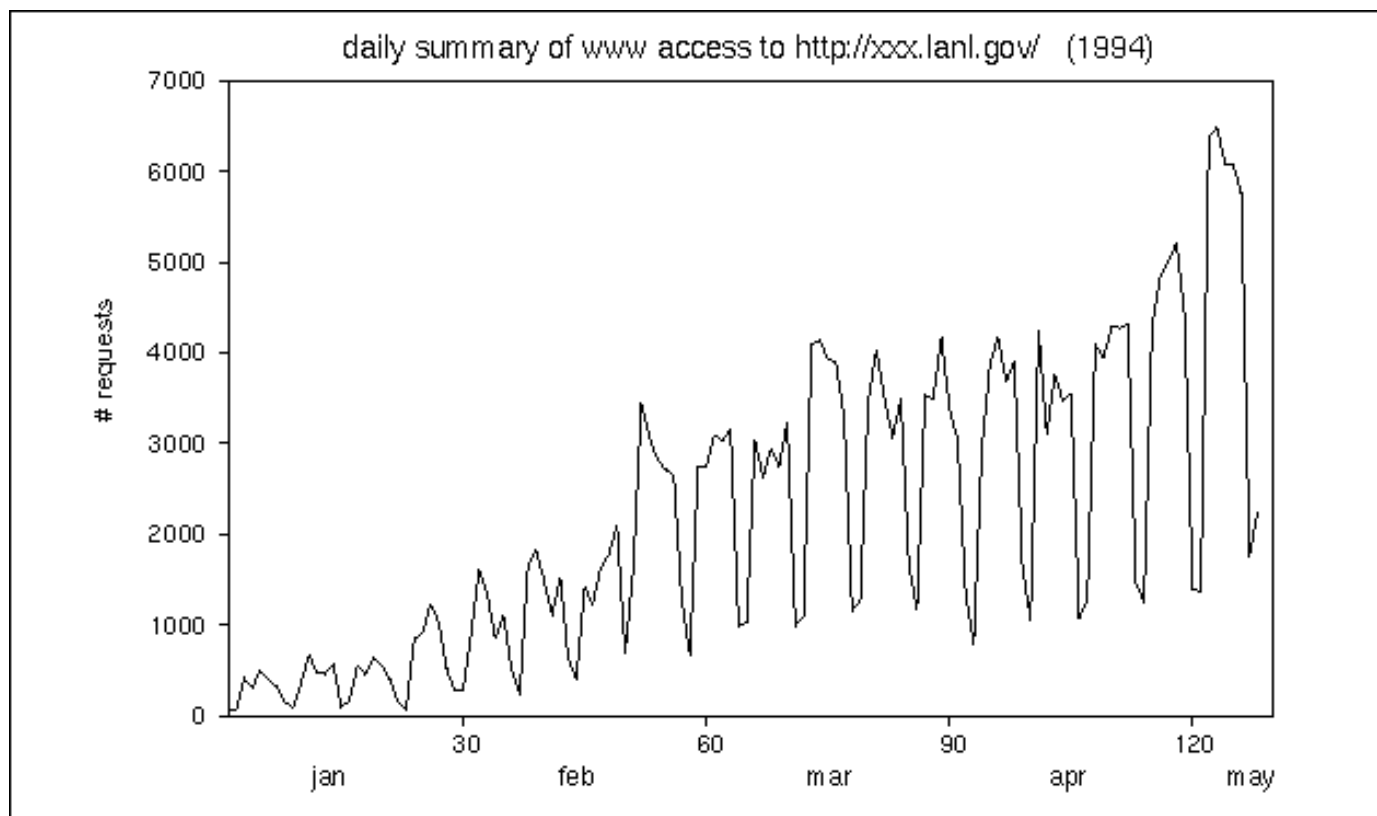


Figure 4. Requests to xxx.lanl.gov through WorldWideWeb

The graph shows the number of http (hypertext transfer protocol) requests to the WorldWideWeb interface on xxx.lanl.gov per day beginning January 1, 1994. The seven-day periodicity is evidence that many physicists still do not have readily available network access on weekends.

recognition of the superior service provided by e-print archives; and third, by the fact that in some of the fields served by e-print archives, it has become customary to provide a paper's electronic-archive index number as a reference rather than a local report number or a published reference.

Prospects and Concerns

The system, in its present form, was not intended to replace journals but only to organize what was once a haphazard and unequal distribution of electronic preprints. It is increasingly used as an electronic journal, however, be-

cause retrieving a copy of a paper electronically is more convenient than physically retrieving a paper from a file cabinet. Aside from minimizing geographic inequalities by eliminating the "boat-mail gap" between continents, the system institutes a form of democracy in research wherein access to new results is granted equally to everyone from beginning graduate students to seasoned operators. No longer is it crucial to have the correct connections or to be on exclusive mailing lists to be kept informed of progress in one's field. The pernicious problem of lost or stolen preprints experienced by some large institutions is also definitively excoriated. The many institutions that

have eliminated their hard-copy distribution of preprints have already seen significant savings in time and money; others have specifically requested that hard copy no longer be sent to them, since electronic distribution has proven reliable and more efficient. Implementing a billing system for the use of an e-print archive would be fairly straightforward; however, such archives cost so little to set up and maintain that they can be offered virtually free. Overburdened terminal resources at libraries are not an issue, since access is typically via the terminal or workstation on one's desk or in the nearest computer room.

Electronic research archives will prove particularly useful for new and

emerging interdisciplinary areas of research for which there are no existing print journals and for which information is consequently difficult to obtain. In many such cases, it is advantageous to avoid a proliferation of premature or ill-considered new journals. Cross-linking of various databases provides an immediate virtual meeting ground for researchers who wouldn't ordinarily communicate with one another. Researchers can quickly establish their own dedicated electronic archive when it is appropriate and ultimately disband if things do not pan out—all with far greater ease and flexibility than is provided by traditional publication media.

Electronic access to scientific research will be a major boon to developing countries, since the expense of connecting to an existing network is infinitesimal compared with that of constructing, stocking, and maintaining libraries. (I frequently receive messages from physicists in developing countries confirming how much better off they find themselves even in the short term with the advent of electronic distribution systems—they are no longer “out of the loop.” Others report feeling that their own research gets a more equitable reading—their research is no longer dismissed for the superficial reasons of low-quality printing or paper stock.) Now that much of the technology has ripened, Eastern European and third-world nations may rapidly develop their electronic infrastructures to the level that took developed nations over a decade to reach—a level at which data-transmission lines are as common as telephone service and terminals and laser-printers as common as typewriters and copy machines. (Similar comments apply equally to the less well-endowed institutions in the U.S., and the changes experienced by physics and biology departments are soon to be repeated by the full range of conventional academic

institutions, including teaching hospitals, law schools, humanities departments, and ultimately public libraries and public grade schools.)

E-print archives will eventually bring great changes to the scientific-journal industry as well. Over the past decade publication companies have been somewhat irresponsible—increasing the number of journals and as well the subscription price per journal (some single journal subscriptions to libraries now run well over \$10,000 per year) during a period when libraries are experiencing a decrease in both funds and space. Publishers have been slow to incorporate electronic communication into their operation and distribution, although such a move would ultimately result in dramatic savings in cost and time for all involved.

Some members of the community have voiced their concern that electronic distribution will somehow increase the number of preprints produced, or encourage dissemination of preliminary or incorrect material. This concern, however, confuses the method of production with the method of distribution—most researchers are *already* producing at saturation. Moreover, once posted to an archive, the electronic form is instantly publicized to thousands of people. Thus the embarrassment over incorrect results is, if anything, *increased*. Such submissions cannot be removed; they can only be replaced by a note that the work has been withdrawn as incorrect, leaving a more permanent blemish than a hard copy of limited distribution that is soon forgotten.

The widespread use of e-print archives does not necessarily make refereed forums obsolete. In some disciplines, the refereeing process plays a useful role in improving the quality of published work, filtering out large amounts of irrelevant or incorrect mate-

rial, and validating research for the purpose of job and grant allocation. A refereeing mechanism could be easily implemented for the e-print archives in the form of either a filter prior to electronic distribution or a review after submission by volunteer readers and/or selected reviewers. In either case, the archives could be partitioned into one or more levels of refereed and unrefereed sectors. Thus, lifting the artificial financial constraints on dissemination of information and decoupling it from the traditional refereeing process will allow for more innovative methods of identifying and validating significant research.

Problems may arise, however, as computer networking spreads outside of the academic community. For example, hep-th would be somewhat less useful if it were to become inundated by submissions from “crackpots” promoting their perpetual-motion machines. It is clear that the architecture of the information highways of the future will somehow have to reimplement the protective physical and social isolation currently enjoyed by ivory towers and research laboratories.

Increased standardization of networking software and electronic storage formats during the 1990s encourages us to fantasize about other possible enhancements to scholarly research communication—in particular, discussion “threads” in which users respond to one another's comments on a specific topic. Usenet newsgroups, for reasons such as their lack of indexing and archiving and their open nature, are unlikely to prove adequate for serious purposes. On the other hand, it is now technically simple to implement a WorldWideWeb form-based submission system to build hyperlinked threads, accessible from given points in individual papers and also started from a subject-based linked discussion page. All posted text could be indexed by the WAIS (Wide Area In-

formation Server) scheme for easy retrieval, and related threads could interleave and cross-link in a natural manner, with standard methods for moving forward and backtracking. A histogram-like interface showing the activity on each thread would facilitate finding threads of current interest, and the index could allow location of all postings by a given person (including self) with the date of latest follow-up to facilitate tracking of responses. This would provide a much more flexible format than Usenet, specifically avoiding awkward protocols for group creation and removal as well as avoiding potentially unscalable aspects of nntp (the network news transfer protocol). For the relatively circumscribed physics research community, a central database (copied onto many nodes, as usual) would have no difficulty with storage or access bandwidth. To enable full-fledged research communication with in-line equations or other linkages, we require slightly higher quality browsers than are currently available. But with hypertext transfer protocols (http) now relatively standardized, network links and links to other application software can be built into underlying TeX documents (and configured into standard macro packages) to be either interpreted by dedicated TeX previewers or passed by a suitable driver into more archival formats (such as Adobe Acrobat PDF) for greater portability across platforms. Multi-component messages could also be assembled in a graphical user interface for composing MIME (multipurpose internet mail extension) messages to be piped to the server by means of the http POST protocol, thereby circumventing some of the inconvenient baggage of Internet sendmail or FTP protocols.

While the above is technically straightforward to implement, there remains the aforementioned issue of lim-

iting access to emulate that effective insulation from unwanted incursions afforded by corridors and seminar rooms at universities and research laboratories. One method would be to employ a “seed” mechanism—that is, to start from a given set of “trusted users” and let them authorize others (and effectively be responsible for those beneath them in the tree), with guidelines such as that the new users must have doctorates or be doctoral candidates, and make permission to post/authorize revocable at any time, retroactive one level back in the tree. To allow global coverage, application to the top level for authorization could be allowed to start a new branch. The scheme entails some obvious compromises, and other schemes are easily envisioned, but the ultimate object remains to determine the optimal level of filtering for input access to maintain an auspicious signal-to-noise ratio for those research communities that prefer to be buffered from the outside world. This would constitute an incipient “virtual communication corridor,” further facilitating useful research communication in what formerly constituted both pre- and post-publication phases, and rendering ever more irrelevant individual researchers’ physical location.

Finally, we mention that the e-print archives in their current incarnation already serve as surprisingly effective inducements for computer literacy, and they have motivated some dramatic changes in computer usage. Researchers who previously disdained computers now confess an addiction to e-mail. Many researchers who for years had refused to switch to UNIX or to TeX are in the process of converting; others have suddenly discovered the power of browsing with World-WideWeb. The system’s effectiveness in motivating these changes justifies the philosophy of providing dual function-

ality in the form of top-of-the-line search, retrieval, and input capabilities for cutting-edge power users, while maintaining “lowest-common-denominator” capabilities for the less “network-fortunate.”

Conclusions and Open Questions

These systems are still primitive, and they represent only tentative first steps in the optimal direction. To summarize, thus far we have learned.

- ▶ The exponential increase in usage of electronic networking over the past few years opens new possibilities for both formal and informal communication of research information.
- ▶ In some fields of science, electronic preprint archives have been on-line since mid-1991 and have become the primary means of communicating research information to many thousands of researchers within the fields they serve. It has been established that people will voluntarily subscribe to receive information from these systems and will make aggressive use of them if they are set up properly. It is anticipated that such systems will grow and evolve rapidly in the next few years.
- ▶ From such experimental systems, we have learned that open (unrefereed) distribution of research information can work well for some disciplines and has advantages for researchers in both developed and developing countries. We have also learned that the technology and network connectivity are currently adequate to support such systems, the performance of which should benefit from the continuing improvements in technology.

I conclude with some unanswered questions to amplify some of my earlier comments:

- ▶ Who will ultimately be the prime beneficiaries of electronic research communication (that is, researchers, publishers, libraries, or other providers of network resources)?
- ▶ What factors influence research communities in their rate and degree of acceptance of electronic technology, and what mechanisms are effective in facilitating such changes?
- ▶ What role will be played by the conventional peer-refereeing process in the electronic media, and how will it differ from field to field?
- ▶ What role will be played by publishing companies, and how large will their profits be? If publication companies do adopt fully electronic distribution, will they pass along the reduced costs associated with the increased efficiency of production and distribution to their subscribers? Can publishing companies provide more value than an unmanned automated system whose primary virtue is instant retransmission?
- ▶ What role will be played by library systems? (Will information be channeled through libraries or, instead, directly to researchers?)
- ▶ How will copyright law be applied to material that exists only in electronic form? At the moment publishing companies are “looking the other way,” living with the dissemination of the electronic preprint information as they did with the earlier preprinted form—claiming that it would be antithetical to their philosophy to impede dissemination of information. Will

they continue to be so magnanimous when libraries begin to cancel journal subscriptions?

- ▶ What storage formats and network utilities are best suited for archiving and retrieving information? Currently we use a combination of e-mail, anonymous FTP, and window-oriented utilities, such as Gopher and WorldWideWeb, combined with WAIS indexing to retrieve TeX and Postscript documents. Will something even better—for example, Acrobat or some other format currently under development—soon merge with the above or emerge as a new standard?
- ▶ How will the medium itself evolve? Conservatively, we can imagine “interactive” journals in which equations can be manipulated, solved, or graphed; citations can instantly open references to the relevant page; comments and errata dated and keyed to the relevant text can be inserted as electronic “post-it notes” in the margins, and so on. Ultimately we will have a multiply interconnected network hypertext system with transparent pointers among distributed databases that transcends the limits of conventional journals in structure, content, and functionality, thereby transforming the very nature of *what* is communicated. These are the kinds of benefits for which we should certainly be willing to pay. Certainly we do not wish to clone current journal formats (determined as they are by the constraints of the print medium) in the electronic medium—we are already capable of distinguishing information content from superficial appearance. Who will decide the standards required to implement any such progress?

This began for me as a spare-time project to test the design and implementation of an electronic preprint distribution system for my own relatively small research community. Its feasibility had been the subject of contentious dispute, and its realization was thought—even by its proponents—to be several years in the future. Its success has led to an unexpectedly enormous growth in usage. It has expanded into other fields of research and has elicited interest from many others—I have received over one hundred inquiries into setting up archives for different disciplines. Each discipline will have slightly different hardware and software requirements, but the current system can be used as a provisional platform that can be tailored to the specific needs of different communities. Despite the success of this project, for three years it remained a spare-time project with little financial or logistical support. Only very recently have the Laboratory, certain government funding agencies, and certain professional societies moved to increase their levels of involvement.

Further development will require coordination among interested researchers from various disciplines, computer and networking staff, and interested library personnel. In particular, it will require dedicated staffing. At the moment, hardware and software maintenance of existing automated archives remains a loosely coordinated volunteer operation, and little further progress can be made on the issues raised by the current systems without some thoughtful direction. Perhaps the centralized databases and further software development will ultimately be administered and systematized by established publishing institutions—if they are prescient enough to reconfigure themselves for the inevitable. Since it has been researchers who have taken the lead thus far, however, we should retain this unique op-

portunity to continue to lead the development of such systems in optimal directions and on terms maximally favorable to ourselves. ■

Acknowledgements

Many people have contributed (consciously or otherwise) to the development of these systems. The original distribution list from which hep-th sprung in 1991 was assembled by Joanne Cohn, whose incipient efforts demonstrated that members of this community were eager for electronic distribution (and Stephen Shenker recommended that the original archive name not include the string "string"). Continual improvements have been based on feedback from users too numerous to credit (although among the most vocal have been Tanmoy Bhattacharya (T-8), Jacques Distler, Marek Karliner, and Paul Mende). People who have administered some of the remote-based archives include Dave Morrison, Bob Edwards, Roberto Innocente, Erica Jen (T-7), and Bob Parks. Joseph A. Carlson (T-5) and David Thomas set up the original Gopher interfaces in late 1992. The Network Operations Center at Los Alamos National Laboratory has reliably and uncomplainingly supplied the requisite network bandwidth @lanl.gov, and Joseph H. Kleczka (C-8) has been available for crisis control. Louise Addis and the staff at the SLAC library moved quickly to incorporate e-print information into the SPIRES database, furthering their decades of tireless electronic service to the high-energy physics community. Dave Forslund (Advanced Computing Laboratory) and Richard Luce (CIC-14) helped lobby for support from within the Laboratory, and the Advanced Computing Laboratory has in addition provided some logistical and moral support. Finally, Geoffrey B. West (T-8) repeatedly and against all obvious reason insisted that the Los Alamos National Laboratory is an appropriate sponsor for this activity, while simultaneously bearing the bad news both from within the Laboratory and from certain government funding agencies



Paul H. Ginsparg received his A.B. in physics from Harvard University in 1977 and his Ph.D. in physics from Cornell University in 1981 under the direction of Kenneth G. Wilson. He then joined the physics department at Harvard University as a Junior Fellow, eventually becoming an Associate Professor. In 1990 he came to the Elementary Particles and Field Theory Group in the Laboratory's Theoretical Division, where he carries out research in relativistic quantum field theory.