JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 16, No. 5, September–October 1993

Nonrecursive Order N Formulation of Multibody Dynamics

Andrew J. Kurdila* and Ramesh G. Menon†
Texas A&M University, College Station, Texas 77843
and
John W. Sunkel‡
NASA Johnson Space Center, Houston, Texas 77058

Although excellent progress has been made in deriving algorithms that are efficient for certain combinations of system topologies and concurrent multiprocessing hardware, several issues must be resolved to incorporate transient simulation in the control design process for large space structures. Specifically, strategies must be developed that are applicable to systems with numerous degreees of freedom. Algorithms are required that induce parallelism on a fine scale, suitable for the emerging class of highly parallel processors. This paper addresses these problems by employing the range space formulation of multibody dynamics and solving for multipliers using the preconditioned conjugate gradient method. By employing regular ordering of the system connectivity graph, an extremely efficient preconditioner can be derived from the range space metric. The method can achieve performance rates that depend linearly on the number of substructures. Furthermore, the approach is promising as a potential parallel processing algorithm in that it exhibits fine parallel granularity, is nonassembling, and is easily load balanced among processors without relying on system topology to induce parallelism.

Introduction

THERE is no doubt that an effective design process for large space structures such as the space station absolutely requires that high-fidelity simulations of the transient response to control inputs be rapidly attainable. Much research has been carried out over the past few years that concentrates on improving the performance of methods for simulating the dynamics of nonlinear, multibody systems.[1-4] The research has been primarily devoted to the derivation of more efficient formulations of multibody dynamics and to the derivation of parallel processing algorithms.

Perhaps the most significant research addressing these two areas has been the introduction of the recursive, order N or $\mathcal{O}(N)$ algorithms[5] and their subsequent refinements for systems of rigid bodies.[4,6] These methods are $\mathcal{O}(N)$ in the sense that the computational cost of the solution procedure is linear in the number of degrees of freedom N of the system,[4] whereas conventional Lagrangian formulations are of cubic order. The conclusion that the Lagrangian methods are of cubic order derives from the fact that a system generalized mass/inertia matrix of dimension $N \times N$ must be factored at each time step. Just as importantly, the computational structure of the recursive $\mathcal{O}(N)$ algorithms is amenable to parallel computation for some system topologies. If the system to be modeled has many independent branches in its system connectivity graph, the computational work required by the algorithm can be distributed among processors by assigning branches to independent processors. Because of the system connectivity and specific hardware architecture, excellent performance improvements and processor utilization are achieved.[6] Because of these successes for rigid-body simulations, it is well-known that many research institutions are investigating adaptations of the original recursive method to ever wider classes of systems. No doubt, the result will be highly efficient algorithms that perform well. Still, three key goals must be resolved before a general parallel processing algorithm can be obtained.

1) Algorithms are required that induce parallelism on a finer scale, suitable for the emerging class of highly parallel processors.

2) Concurrent transient simulation methods must be automatically load balancing for a wider collection of combinations of mechanical systems and concurrent multiprocessing hardware.

3) The transient simulation method should also be amenable to vector processing implementation on each independent concurrent multiprocessor.

These goals should be very challenging if the algorithm is based upon a recursive $\mathcal{O}(N)$ formulation.

An innovative strategy based on the stated goals is derived in this paper. In part, its foundation can be traced to element-by-element methods already in use in finite-element solution procedures.[7] The combination of the range space formulation with the preconditioned conjugate gradient solution results in an extremely efficient sequential algorithm for a class of problems described in the paper. The efficiency is primarily due to the selection of a block Jacobi preconditioner that is rapidly convergent. The method is nonassembling, and as a result, it does not require a large amount of in-core storage. Consequently, it is also attractive as a candidate for implementation on workstations. Current studies, described herein, indicate that due to the rapid convergence achieved by using the selected preconditioner, the method can achieve performance rates that depend linearly upon the number of substructures.

The method should be readily implementable on parallel processors since a vast literature exists on the amenability of the preconditioned conjugate gradient solution procedure to both concurrent and vector processing. The method is relatively easily load balanced among processors and does not rely upon system topology to induce parallelism.

This paper focuses on the fundamental dynamical formulation using a combination range space/preconditioned conjugate gradient solution, and its performance on sequential computing machines.[8] Although the potential application of the method on parallel architectures is outlined, the details of

Presented as Paper 91-1112 at the AIAA/ASME/ASCE/AHS/ASC 32nd Structures, Structural Dynamics, and Materials Conference, Baltimore, MD, April 8-10, 1991; received July 23, 1991; revision received Sept. 23, 1992; accepted for publication Oct. 6, 1992. Copyright © 1991 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Assistant Professor, Department of Aerospace Engineering. Member AIAA.

†Graduate Student, Department of Aerospace Engineering. Student Member AIAA.

‡Aerospace Engineer, Navigation, Control and Aeronautics Division. Associate Fellow AIAA.

a concurrent implementation are presented in a forthcoming paper.[9]

Governing Equations

The range space formulation of dynamics has been derived in the aerospace and mechanism dynamics research literature in Refs. 10–12. Its theoretical foundation can be traced to the range space formulation of constrained quadratic optimization.[13] Despite the fact that it is often less computationally expensive than the null-space methods, the null-space method seems to have received more attention in the literature.[14–17]

The dynamics of a nonlinear, multibody system are governed by the collection of differential-algebraic equations

$$M(q)\ddot{q} = f(q,\dot{q},t) + \left[C(q)\right]^T\lambda \tag{1}$$

subject to constraints in linear, nonholonomic form

$$C(q)\dot{q} = 0 \tag{2}$$

where q is the vector of generalized coordinates of length N, M the generalized matrix of inertias, and λ is the vector of Lagrange multipliers of length d. When the constraints are holonomic, C is the constraint Jacobian matrix. If Eq. (2) is differentiated with respect to time t, an expression involving the second derivatives of the generalized coordinates is obtained as follows:

$$C\ddot{q} = -\frac{dC}{dt}\dot{q} \equiv e(q,\dot{q},t) \tag{3}$$

The range space solution is found by explicitly solving for the multipliers λ using Eqs. (1) and (3)

$$(CM^{-1}C^T)\lambda = -CM^{-1}f + e \tag{4}$$

We refer to the coefficient matrix of λ in Eq. (3) as the constraint metric. Substituting for λ in Eq. (1) results in a system of ordinary differential equations given by

$$\ddot{q} = M^{-1}\left[f - C^T(CM^{-1}C^T)^{-1}(CM^{-1}f - e)\right] \tag{5}$$

Any standard explicit-predictor/implicit-corrector or Runge-Kutta integration scheme can be applied to these equations provided that the condition number of the constraint metric does not become too large. The condition number of a matrix A is given by

$$\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \tag{6}$$

The restriction that the condition number remains small precludes the possibility of redundant constraints (for example, as associated with singularities arising from closed loops) and remains an underlying assumption throughout the rest of the paper.

One advantage of the range space equations for systems having many independent structures to be assembled is that the system coefficient matrix is block diagonal. Consequently, the factorization and back substitution required to solve a system of equations involving the mass matrix is relatively inexpensive. It requires that one calculate the factorization of the individual substructure mass matrices alone as opposed to the factorization of the system mass matrix. In fact, one need not even assemble the system mass matrix, and the factorizations can occur in parallel. Unfortunately, if one subdivides the overall system into finer collections of substructures (to facilitate the factorization of the system coefficient matrix), numerous constraints are introduced into the model.

The approach taken in this paper is to finely subdivide the system to be modeled and thus accrue the benefits of having a system coefficient matrix with smaller block diagonals, but also employ a solution procedure that ameliorates the cost associated wih the increasing dimensionality of the constraint metric. Specifically, the solution for the Lagrange multipliers in Eq. (3) is carried out using the preconditioned conjugate gradient procedure.

Preconditioned Conjugate Gradient Solution

The preconditioned conjugate gradient procedure is an "accelerated" variant of the classical conjugate gradient procedure.[18] The preconditioned conjugate gradient algorithm to solve the linear system of equations

$$Ax = b \tag{7}$$

is given in Table 1. Careful inspection of the algorithm shows that the most computationally expensive tasks in the procedure are the calculation of the product of the coefficient matrix A and a given residual vector and a solution of a linear system of equations requiring the factorization of the preconditioner Q.

The rate of convergence of the preconditioned conjugate gradient algorithm is accelerated by employing a user defined "preconditioning matrix." This matrix must have two properties to be an effective preconditioner:

1) It must be relatively easy to factor.

2) It must be an approximate inverse of the constraint metric in a sense to be made precise in the following.

The reason for employing the preconditioned conjugate gradient solution method is that the convergence rate of the conjugate gradient algorithm (that is, with Q being the identity matrix) is governed by

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \le \left[\frac{1-\sqrt{\kappa(A)}}{1+\sqrt{\kappa(A)}}\right]^{2k} \tag{8}$$

Thus, the rate of convergence of the algorithm improves as the condition number of A, $\kappa(A)$, decreases. The convergence of the preconditioned conjugate gradient method is governed by the same expression,[18] but with A replaced by

$$Q^{-1/2}AQ^{-1/2} \tag{9}$$

Clearly, if the preconditioner is identical to the coefficient matrix A, then the condition number is minimized. Hence, the preconditioner sought should be such that its inverse approximates the inverse of the coefficient matrix. Many methods exist for the calculation of preconditioners. It should be noted that whereas the motivation for the use of many of these preconditioners is mathematically sound, the final choice invariably involves some heuristic.

The choice of the preconditioners employed in this paper is based upon the following assumptions regarding the structural/mathematical system to be modeled:

1) The system closely resembles a series of chains of bodies.

2) The number of interface degrees of freedom is small relative to the number of interior degrees of freedom for a substructure.

Table 1 Preconditioned conjugate gradient algorithm

Initial guess x_0
  $r_0 = b - Ax_0$
  for k = 1 ··· max, where max is the maximum number of iterations allowed
    If $r_{k-1} = 0$
      $x = x_{k-1}$, exit loop!
Else
  Solve $QZ_{k-1} = r_{k-1}$, where Q is the preconditioner matrix
  $\beta_k = \langle z_{k-1}, r_{k-1}\rangle / \langle z_{k-2}, r_{k-2}\rangle$
  $p_k = z_{k-1} + \beta_k p_{k-1}$
  $\alpha_k = \langle z_{k-1}, r_{k-1}\rangle / \langle p_k, Ap_k\rangle$
  $x_k = x_{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k Ap_k$

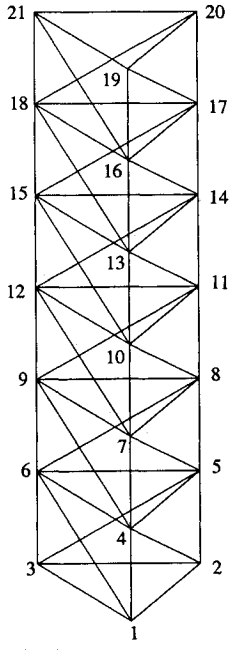

Fig. 1 63-degrees-of-freedom Z-truss substructure.

3) The system does not contain any closed chains.

To a large extent, these assumptions have been motivated by the physical structure of the space station in its assembly complete configuration.

The preconditioner for the system constraint metric, $CM^{-1}C^T$, is based on the topology of a chain of substructures. If there are k constraint interfaces with d_i constraints at the ith interface, the constraint Jacobian matrix connecting two bodies at the ith interface can be denoted as

$$[C_i(q)] \in R^{d_i \times N} \tag{10}$$

The system constraint Jacobian matrix has the form

$$[C(q)] = \begin{bmatrix} [C_1] \\ [C_2] \\ \cdots \\ [C_k] \end{bmatrix} \tag{11}$$

The system constraint metric can then be written as

$$\begin{bmatrix} [C_1][M]^{-1}[C_1]^T & [C_1][M]^{-1}[C_2]^T & \cdots & [C_1][M]^{-1}[C_k]^T \\ [C_2][M]^{-1}[C_1]^T & [C_2][M]^{-1}[C_2]^T & \cdots & [C_2][M]^{-1}[C_k]^T \\ \vdots & \vdots & & \vdots \\ [C_k][M]^{-1}[C_1]^T & [C_k][M]^{-1}[C_2]^T & \cdots & [C_k][M]^{-1}[C_k]^T \end{bmatrix} \tag{12}$$

Based on the structure of the constraint metric [Eq. (12)], the preconditioner is selected to be the block diagonal matrix

$$\begin{bmatrix} [C_1][M]^{-1}[C_1]^T & & & \\ & [C_2][M]^{-1}[C_2]^T & & \\ & & \ddots & \\ & & & [C_k][M]^{-1}[C_k]^T \end{bmatrix} \tag{13}$$

Although the off-diagonal blocks

$$[C_i][M]^{-1}[C_j]^T \quad \text{for} \quad i \neq j \tag{14}$$

are not generally identically equal to zero, this choice of preconditioner is shown to be extremely efficient for the class of problems described in the next section. Furthermore, this preconditioner satisfies the two essential criteria of good preconditioners.

1) It is block diagonal, with small diagonal blocks, and, as a result, is relatively easy to factor.

2) It has an inverse that provides a good approximation to the inverse of the full system coefficient matrix.

This latter conclusion results from the well-known fact[19] that the directed graph representing the connectivity of an open-loop system can be regularly ordered. The regular ordering results in a system constraint metric that has a reduced bandwidth. That is, many of the off-diagonal blocks are identically equal to zero for $i \gg j$. The choice of the preconditioner shown is often called the block Jacobi preconditioner and is known to be highly effective for classes of systems of equations arising from elliptic partial differential equations.

Example Simulations

For purposes of illustration, only linear simulations are considered herein. Similar results for nonlinear multibody systems have been presented.[9,20] To establish the efficiency and performance of the combined range space/preconditioned conjugate gradient algorithm, several transient simulations have been carried out. The first such simulation has been designed to answer two key questions regarding the feasibility of the approach for multibody structures:

1) How efficient is the selected preconditioner and how does the preconditioner improve the effectiveness of the conjugate gradient iteration processes as a whole?

2) How efficiently can the transient response be carried out? In particular, what is the computational cost of solving the constraint metric factorization at each time step?

These two questions, that is, preconditioner efficiency and constraint metric factorization were judged to the crucial computational bottlenecks for the algorithm as a whole.

The two substructures selected for evaluation of the algorithm in studying its feasibility are shown in Figs. 1 and 2. Figure 1 shows a 63-degrees-of-freedom Z-truss substructure comprised of rod elements generated by the finite element modeling program, MSC PAL. The second assembly, shown in Fig. 2, is a 54-degrees-of-freedom X-frame substructure comprised of three-dimensional beam elements generated by the same program. Figures 3 and 4 illustrate that a remarkable

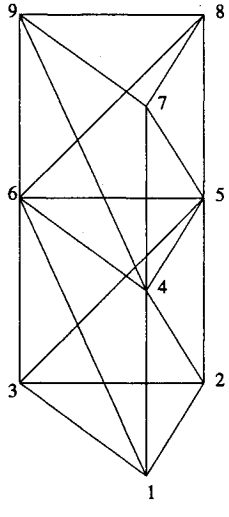

Fig. 2 54-degrees of-freedom X-frame substructure.

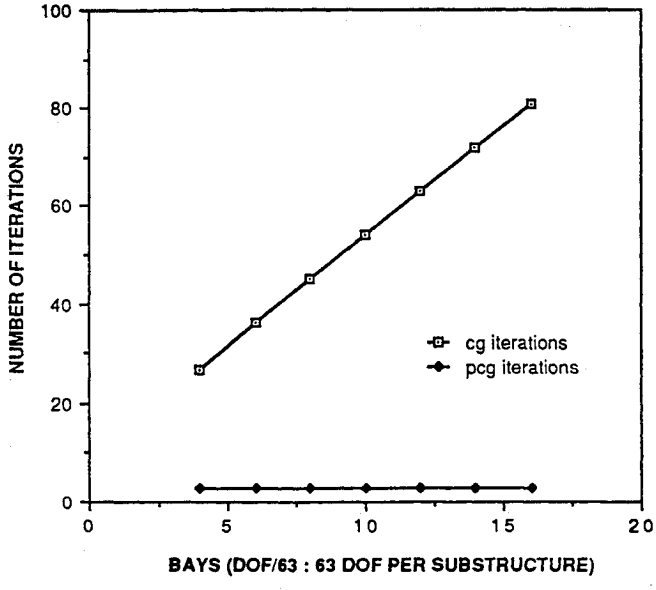

Fig. 3 Iterations as a function of number of degrees of freedom for Z-truss structure.

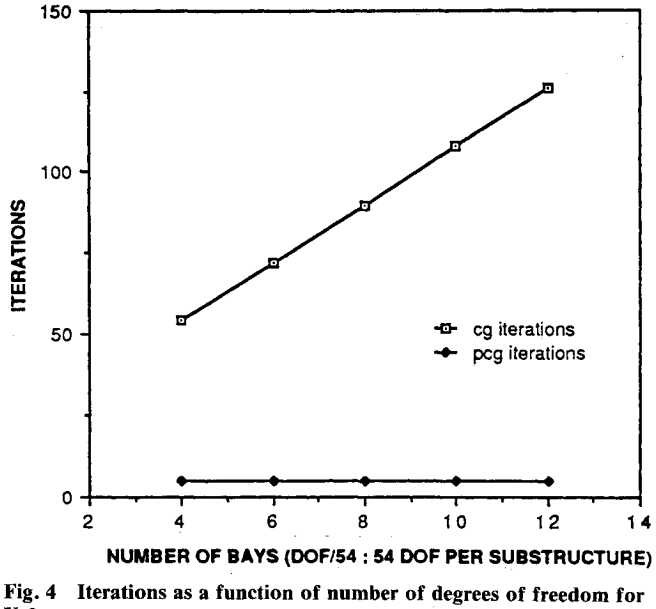

Fig. 4 Iterations as a function of number of degrees of freedom for X-frame structure.

convergence rate is achieved using the preconditioner derived from the regularly ordered graph of the constraint metric. In Fig. 3 the number of iterations required by the preconditioned conjugate gradient algorithm to converge to a tolerance of 1e−14 of the true solution is plotted versus the number of degrees of freedom on the horizontal axis. Thus, the number of flexible degrees of freedom on the horizontal axis varies from 126 to 504 as substructures are cantilevered end-to-end. As clearly illustrated in the diagram, the number of preconditioned conjugate gradient iterations remains constant, and equal to three independent of the number of total degrees of freedom. The other line plotted on the graph is the analytical upper limit to the number of iterations required for the conjugate gradient method. Completely analogous results are depicted in Fig. 4. In this case the number of total flexible degrees of freedom in the structure varies from 108 to 324, and the number of iterations of the preconditioned conjugate gradient algorithm required to achieve convergence to a tolerance of 1e−14 is plotted on the vertical axis. Again, independent of the number of degrees of freedom, the number of iterations remains constant and equal to four. Consequently one can conclude that the algorithm for generating the preconditioner is indeed extremely efficient for the test problems simulated.

Figures 5 and 6 depict the total time required per time step to solve the "inversion" of the system constraint metric. Figure 5 depicts the corresponding results for the Z-truss, whereas Fig. 6 depicts the results for the X-frame. In both cases it is clear that the simulation time grows linearly as a function of the total number of flexible degrees of freedom. To the authors' knowledge, no such result for flexible bodies has been presented to date.

The merits of the approach described in this paper have been demonstrated using the Z-truss and X-frame substructures. This section presents corresponding results for a 13 substructure model of the space station. An early version of the space station in its assembly complete configuration is shown in Fig. 7. For purposes of demonstrating the proposed methodology, the solar panel sections and the habitation modules have not been modeled in the numerical simulations. Figure 8 shows the decomposition of the model into 13 constituent substructures, with each substructure showing its relative location in the space station and the number of degrees of freedom. The substructures are modeled using beam elements and, as a result, each node has six degrees of freedom. Figure 9 shows the CPU time required per time step of simulation as more substructures are added. As demonstrated for the previous cases, the simulation time does grow as a linear function of the number of degrees of freedom. Figure 9 illustrates the impor-

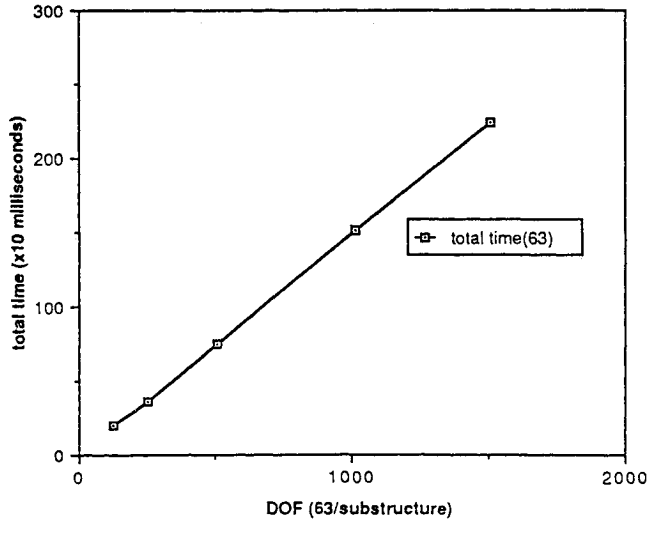

Fig. 5 CPU time per time step for factorization of the constraint metric for Z-truss.

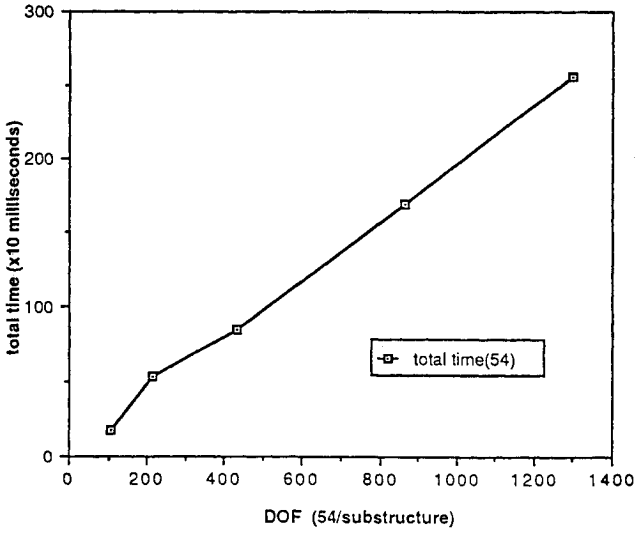

Fig. 6 CPU time per time step for factorization of the constraint metric for X-frame.

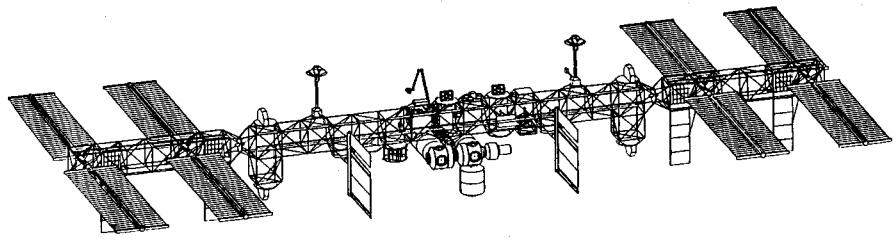

Fig. 7 Schematic of space station model.

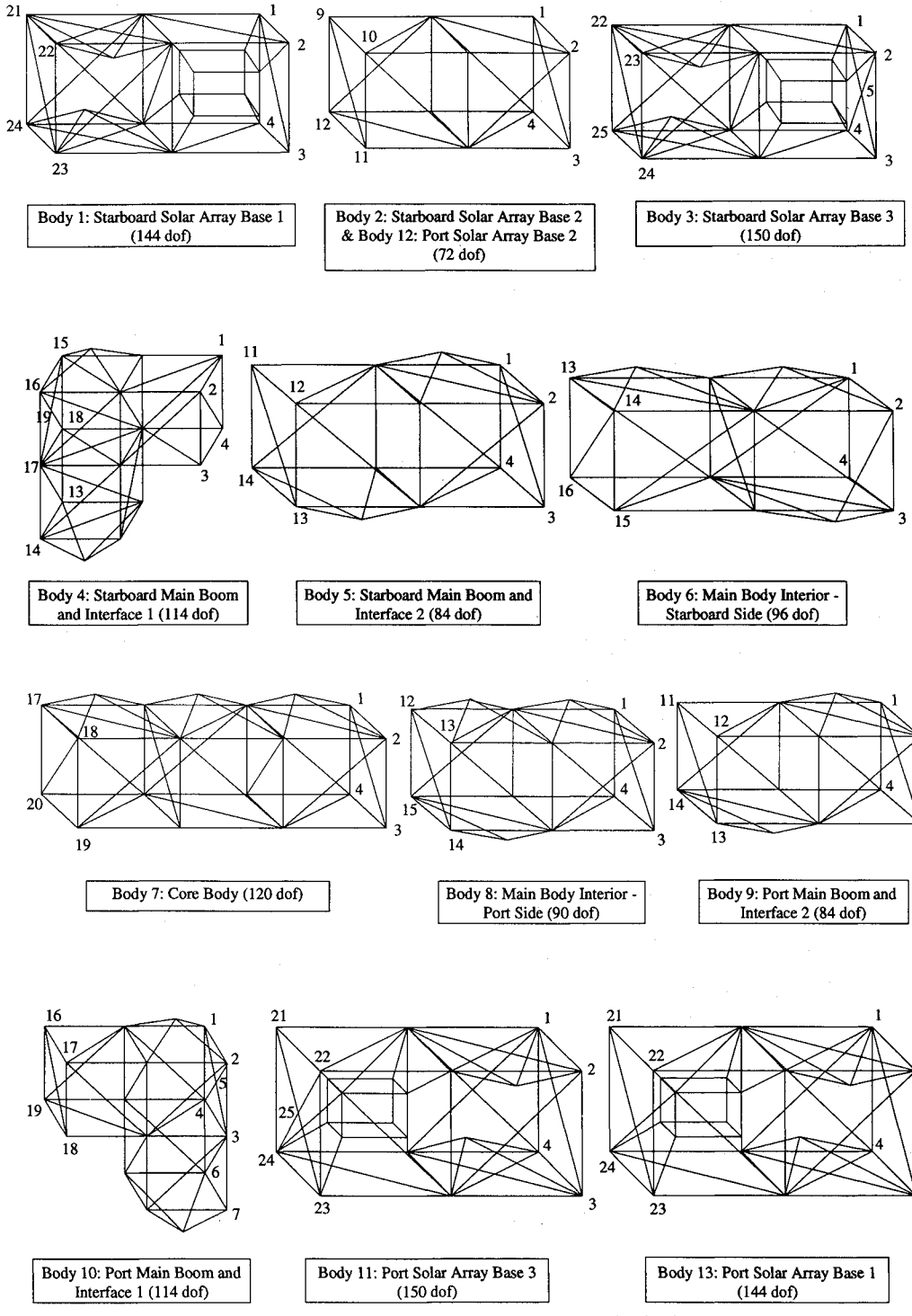

Fig. 8 Decomposition of space station model into 13 substructures.

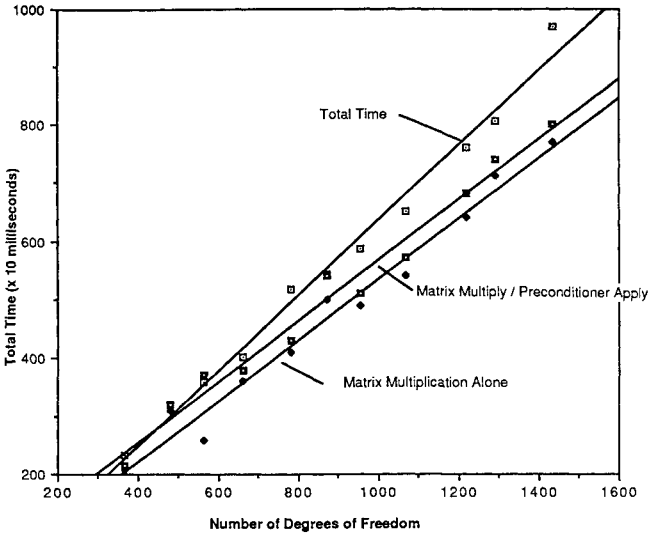

Fig. 9 Computational cost (CPU time) breakdown for space station model.

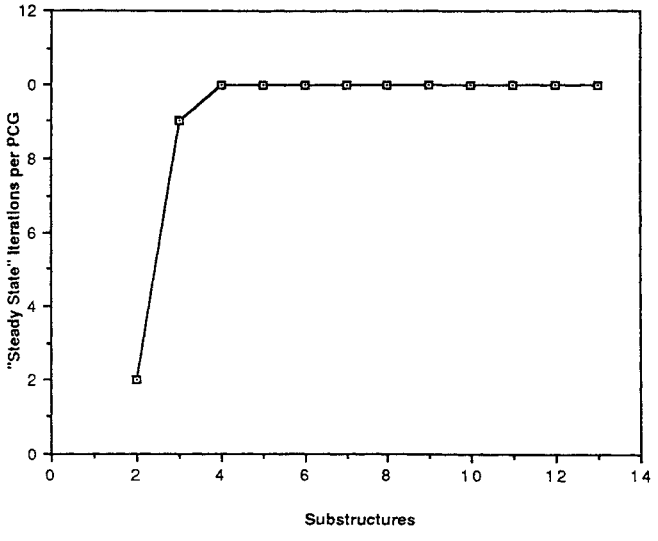

Fig. 10 Iterations as a function of number of substructures.

tant fact that, again, the primary computational cost of the algorithm is associated with 1) system matrix multiplication and 2) application of the preconditioner.

Both of these operations are parallelizable with little effort due to their block diagonal structure. Figure 10 shows that the number of preconditioned conjugate gradient iterations required for convergence is nearly independent of the number of substructures or degrees of freedom employed in the model.

Potential for Parallelism

As noted earlier, most work to date in deriving concurrent algorithms for multibody simulation have been based on the recursive $\mathcal{O}(N)$ algorithm. Typically, concurrence in these methods has been induced by assigning independent branches in the connectivity graph of the system to independent processors. It is important in evaluating these methods to realize their inherent attributes that complicate efforts to derive general concurrent multiprocessing algorithms.

1) These methods depend upon system topology to induce parallelism.

2) These methods induce parallelism on a large scale or in a "large granularity."

Now, for a subclass of present-day computer architectures and a subclass of systems to be simulated, these approaches are extremely efficient. For example, hardware configured with a few independent processors such as are available from Cray: ~4 processors, Flex: ~8 processors, and Alliant: ~8 processors can be employed easily with the recursive, $\mathcal{O}(N)$ algorithms. Likewise, systems that have a matching number of independent branches will provide good examples of performance and processor utilization. The trend in parallel computing architectures is toward more independent processors, potentially with vector processor capabilities. Recent examples include BBN Butterfly: ~32 processors, N-Cube: ~32+ processors, and Capps 9064: ~32 processors.

Strategies that induce parallelism by assigning independent branches of the mechanical system to independent processors currently fail to make full use of the computing power of these latter architectures. Furthermore, many structures of interest today, such as the space station, seldom exhibit such redundant topology.

The concurrent multiprocessing strategy proposed for the range space/preconditioned conjugate gradient algorithm has the following advantageous features:

1) It induces parallelism on a fine scale, suitable for the new class of processors.

2) It is essentially load balancing.

3) The method can use existing symbolic equation generation codes to generate appropriate, efficient substructure equations.

4) The preconditioned conjugate gradient solution has well-documented amenability to concurrent and vector processing techniques. The block diagonal structure of the system coefficient matrix and preconditioner make this even more apparent.

Conclusions

Although the recursive, $\mathcal{O}(N)$ multibody dynamics formulations can yield excellent performance in many simulations, they are not a panacea as regards applications to all classes of problems in multibody dynamics. The benefits of the method for simulating systems with thousands of degrees of freedom, such as the space station, have yet to be firmly established. An alternative nonrecursive, $\mathcal{O}(N)$ algorithm has been presented in this paper that has many advantages for a sequential computing environment including the fact that it is rapidly convergent, it can achieve $\mathcal{O}(N)$ computational cost and is nonassembling. In addition, the method exhibits fine parallel granularity suitable for the emerging computer architectures and is easily load balanced among a large number of multiprocessors. The results of the parallel implementation are deferred to a future paper.[9] Although the examples have been limited to linear systems, the formulation is applicable to nonlinear, multibody simulations.[20]

References

[1]Chiou, J.-C., "Constraint Treatment Techniques and Parallel Algorithms for Multibody Dynamic Analysis," Center for Space Structures and Controls, Univ. of Colorado, CU-CSSC-90-26, Boulder, CO, Nov. 1990.

[2]Gluck, R., "Hope for Simulating Flexible Spacecraft," *Aerospace America,* Nov. 1986, pp. 40–44.

[3]Haug, E. J., "Elements and Methods of Computational Dynamics," *Computer Aided Analysis and Optimization of Mechanical System Dyanmics,* Springer-Verlag, Berlin, 1984, pp. 3–37.

[4]Singh, R. P., Schubele, B., and Sunkel, J. W., "Efficient Algorithms for the Dynamics of Multi-Link Mechanisms," AIAA Paper 89-3527-CP, 1989.

[5]Hollerbach, J. M., "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 10, 1980, pp. 730–736.

[6]Bae, D., and Haug, E., "A Recursive Formulation for Constrained Mechanical System Dynamics," *Mechanics of Structures and Ma-*

chines, Vol. 15, No. 3, 1987, pp. 359–382.

[7]Hughes, T. J. R., *The Finite Element Method,* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[8]Kurdila, A. J., "A Nonrecursive $\Theta(N)$ Preconditioned Conjugate Gradient/Range Space Formulation of MDOF Dynamics," NASA/American Society for Engineering Education Summer Faculty Fellowship Program, NASA Johnson Spaceflight Center, Final Rept. NGT-44-005-803, Houston, TX, Aug. 1990.

[9]Menon, R. G., and Kurdila, A. J., "Subdomain Decomposition Methods and Computational Controls for Multibody Dynamical Systems," *Computing Systems in Engineering,* Vol. 3, Nos. 1–4, 1992, pp. 189–200; see also AAS/AIAA Spaceflight Mechanics Meeting, AAS Paper 91-111, Houston, TX, Feb. 1991.

[10]Agrawal, O. P., "Dynamic Analysis of Multi-Body Systems Using Tangent Coordinates," *The Theory of Machines amd Mechanisms,* edited by J. Garcia-Lomas, Pergamon Press, Oxford, England, UK, 1987, pp. 533–536.

[11]Placek, B., "Contribution to the Solution of the Equations of Motion of the Discrete Dynamical System with Holonomic Constraints," *The Theory of Machines and Mechanisms,* edited by E. Batista, J. Garcia-Lomas, and A. Navarro, Pergamon Press, Oxford, England, UK, 1987, pp. 379–382.

[12]Kurdila, A. J., and Kamat, M. P., "Concurrent Multiprocessing Methods for Calculating Nullspace and Range Space Bases for Multibody Simulation," *AIAA Journal,* Vol. 28, No. 7, 1990, pp. 1224–1232.

[13]Gill, P. E., Murray, W., and Wright, M. H., *Practical Optimization,* Academic Press, London, England, UK, 1981.

[14]Singh, R. P., and Likins, P. W., "Singular Value Decomposition for Constrained Dynamical Systems," *Journal of Applied Mechanics,* Vol. 52, Dec. 1985, pp. 943–948.

[15]Wehage, R. A., and Haug, E. J., "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems," *ASME Journal of Mechanism Design,* Vol. 104, Jan. 1982, pp. 242–255.

[16]Kim, S. S., and Vanderploeg, M. J., "QR Decompostion for State Space Representation of Constrained Mechanical Dynamic Systems," *Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 108, June 1986, pp. 108–118.

[17]Huston, R. L., "Useful Procedures in Multi-body Dynamics," *Dynamics of Multi-body Systems,* edited by G. Bianchi, Springer-Verlag, Berlin, 1986, pp. 69–78.

[18]Golub, G. H., and Van Loan, C. F., *Matrix Computations,* Johns Hopkins University Press, Baltimore, MD, 1983.

[19]Wittenburg, J., *Dynamics of Systems of Rigid Bodies,* B. G. Teubner, Stuttgart, Germany, 1977.

[20]Menon, R. G., and Kurdila, A. J., "A Non-Recursive Order N Method for Simulation of Nonlinear Multibody Systems," First U.S. National Congress on Computational Mechanics, Chicago, IL, July 1991.

Recommended Reading from Progress in Astronautics and Aeronautics

Viscous Drag Reduction in Boundary Layers

Dennis M. Bushnell and Jerry N. Hefner, editors

This volume's authoritative coverage of viscous drag reduction issues is divided into four major categories: Laminar Flow Control, Passive Turbulent Drag Reduction, Active Turbulent Drag Reduction, and Interactive Turbulent Drag Reduction. It is a timely publication, including discussion of emerging technologies such as the use of surfactants as an alternative to polymers, the NASA Laminar Flow Control Program, and riblet application to transport aircraft. Includes more than 900 references, 260 tables and figures, and 152 equations.

1990, 530 pp, illus, Hardback • ISBN 0-930403-66-5
AIAA Members $59.95 • Nonmembers $75.95 • Order #: V-123 (830)

Place your order today! Call 1-800/682-AIAA

AIAA

American Institute of Aeronautics and Astronautics

Publications Customer Service, 9 Jay Gould Ct., P.O. Box 753, Waldorf, MD 20604
FAX 301/843-0159 Phone 1-800/682-2422 9 a.m. - 5 p.m. Eastern

Sales Tax: CA residents, 8.25%; DC, 6%. For shipping and handling add $4.75 for 1-4 books (call for rates for higher quantities). Orders under $100.00 must be prepaid. Foreign orders must be prepaid and include a $20.00 postal surcharge. Please allow 4 weeks for delivery. Prices are subject to change without notice. Returns will be accepted within 30 days. Non-U.S. residents are responsible for payment of any taxes required by their government.