Journal of Cellular Biochemistry 80:181–186 (2000)

Informatics Issues in Large-Scale Sequence Analysis: Elucidating the Protein Kinases of *C. elegans*

Jonathan Bingham, Greg D. Plowman, and Sucha Sudarsanam*

SUGEN, Inc., South San Francisco, California 94080

Abstract With the availability of the nearly complete genomic sequence of *C. elegans*, the first multicellular organism to be sequenced, molecular biology has definitely entered the postgenomic era. Annotation of the genomic sequence, which refers to identifying the genes and other biologically relevant sections of the genome, is an important and nontrivial next step. A first-pass annotation will be necessarily incomplete but will drive further biological experiments, which in turn will help to annotate the genome better. Given the scale of the genome sequence analysis, it is clear that the annotation should be automated as much as possible without sacrificing the quality of analysis. In this work, we outline our approach to identifying the protein kinases of *C. elegans* from the genomic sequence. We describe new tools we have developed for analysis, management and visualization of genomic data. By developing modular and scalable solutions, this study has provided a framework for future analysis of the *Drosophila* and human genomes. J. Cell. Biochem. 80:181–186, 2000.

Recent genomic sequencing efforts are yielding unprecedented volumes of genomic data at a pace previously undreamed of, promising to soon provide several complete genomes, including *Drosophila* and human. The challenge now is to manage, analyze, annotate, and visualize genomic data. The recently completed *C. elegans* genome [The *C. elegans* Sequencing Consortium, 1998] provides a useful glimpse into the prospects and perils of large-scale sequence analysis. At 100 million bases in length, *the C. elegans* genome is still small compared to those of higher eukaryotes, but already large enough to present significant challenges in scalability of analysis. Also, as the first multicellular organism to be completely sequenced, it presents a reasonable model system for the analysis of higher eukaryotes.

For the present purposes, we focused our analysis on identifying protein kinases, which are involved in signal transduction and specifically protein phosphorylation. In eukaryotes, protein phosphorylation plays a critical role in cell cycle regulation, DNA replication, gene transcription, protein translation, and energy metabolism. In multicellular eukaryotes, such as worms, it also affects more complex functionality (such as cellular differentiation, intercellular communication, cell survival and senescence, synaptic transmission, and environmental interaction). Protein kinases constitute 2.4% of the worm genome and are the second most prevalent protein family in worms. A comprehensive analysis of the *C. elegans* protein kinases appears elsewhere [Plowman et al., 1999]; our emphasis here is on describing the computational framework used for the analysis. Our analysis strategy can be applied to other protein families as well as other genomes to identify biologically interesting features.

Data Analysis Pipeline

In order to cope with the huge amount of genomic data, sequence analysis clearly must be automated to the greatest extent possible, while preserving the quality, accuracy, and precision of the results. Relational database management systems greatly simplify the task of data warehousing and retrieval, but creating a central, ordered repository of sequence data is only the beginning. The larger questions are what data to store, how to analyze and annotate it, and how to ultimately visualize the results. Scripting and automation languages make it possible to create a large-scale data flow pipeline with minimal human intervention and human error. It should be emphasized

*Correspondence to: Sucha Sudarsanam, SUGEN, Inc., 230 East Grand Avenue, South San Francisco, CA 94080. E-mail: sucha-sudarsanam@sugen.com

Received 19 July 1999; Accepted 19 July 1999

that these are only enabling technologies that leave open the question of what to automate and how to ensure the quality of the end product. A variety of algorithms exist for homology and profile searching of large data sets, including BLAST [Altschul et al., 1997], Smith-Waterman [Smith and Waterman, 1981], and Hidden Markov Models [Krogh et al., 1994]. These tools form an essential part of a sequence analysis pipeline. But their integration and the interpretation of their results pose a significant challenge. Finally, after constructing a data analysis pipeline, complete with data warehousing, scripting, and algorithms, some new set of annotation data emerges. Ultimately a human must inspect or sift through the data in order to confirm its validity and utility. Existing tools are already of use in data visualization, including genome browsers and tree viewers. But the sheer volume of data already strains many existing visualization tools to their practical limits, while introducing new problems of navigation. In the course of our analysis of the *C. elegans* protein kinases, we found it necessary to create our own data visualization tools. In what follows, we outline our data analysis pipeline.

Data Management

The complete *C. elegans* genome is available for download from the Sanger Center (http://www/sanger.ac.uk), as cosmids, as six chromosomal assemblies, and as 19,099 putative worm proteins. After downloading each data set, we imported the raw sequence data into a relational database, storing the accession numbers, sequence text, time stamp, and so forth. At each step of the analysis, results—either intermediate or final—were imported into new tables, each linked back to the raw sequence data. The Sugen *C. elegans* database, called SuCeDb, contains tables for raw sequence data, for open reading frame predictions, for domain predictions including protein kinases, and for additional sequence annotation (Fig. 1). A typical step in the analysis pipeline requires querying the database, running some form of analysis on the data, and storing the results of the analysis back in SuCeDb. Depending on the complexity of a particular analysis algorithm, database interaction may itself be the rate-limiting step. This is the case, for example, with sequence translation, a trivial enough operation taking on the order of a few minutes

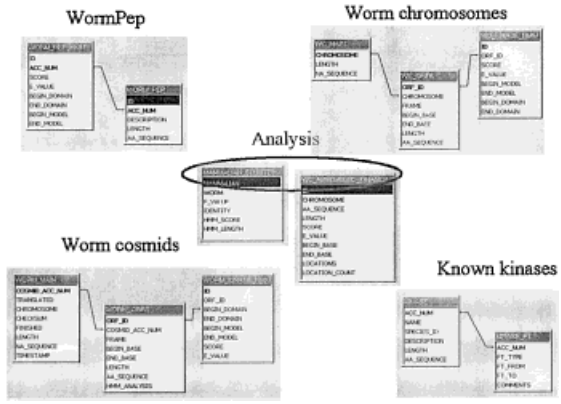

Fig. 1. Schema for sequence analysis dataflow. Database contains tables for raw sequences (Worm chromosomes, Worm cosmids); predicted proteins (WormPep); known kinases and boundaries for catalytic domains (Known kinases); and tables for storing results of the analysis (Analysis).

for the complete worm genome. In the case of more complex analysis, such as homology and pattern-based searching, database access occupies a less significant portion of the overall time. Better hardware and relational database performance tuning can alleviate some of the stress on computer resources. But the biologically interesting portion of the data analysis pipeline is not the storage mechanism, but rather the data and the analysis that produces it.

Analysis

For the nucleic acid databases, the next step after downloading and data warehousing was to translate the sequences into all six reading frames, identifying potential coding regions. All open reading frames (ORFs) longer than 30 amino acids were retained and stored in a separate database table. A variety of algorithms exist for searching and analyzing the opening reading frames, including gene recognition algorithms (for a comprehensive collection of gene prediction algorithms, see http://linkage.rockefeller.edu/wli/gene/), homology search methods [Altschul et al., 1997; Smith and Waterman, 1981], profile search methods [Krogh et al., 1994; Eddy, 1998], and pattern search tools [Henikoff et al., 1999]. As always, the choice of algorithms must depend on the specific nature of the problem. Clearly to identify protein kinases it was not necessary to assemble all genes or identify all intron/exon boundaries in the entire genome.

One search strategy, and by far the most prevalent today, is homology searching, using

TABLE I.

Frame	Begin base	End base	Begin model	End model	E value	Score
−1	21,512	20,169	122	261	4.60E-37	137.2
−2	21,532	21,398	94	122	3.60E-02	14.8
−3	21,648	21,529	71	90	9.40E-02	13.3
−3	21,798	21,652	28	67	1.00E-01	13.2
−2	22,223	21,765	1	21	3.90E-04	21.7

Alignment of ZC404 kinase on the hidden Markov model (HMM). Coordinates for open reading frames FRAME, BEGIN_BASE, END_BASE) that contain the putative kinase exons are shown in the first three columns. Alignments of each exon on the HMM (BEGIN_MODEL, END_MODEL) are shown in the next two columns. Finally, for each kinase exon, expectation values and scores against the model are shown.

either the fast, heuristic BLAST2 algorithm or the slower but more sensitive Smith-Waterman algorithm. These general purpose algorithms accept a query sequence as an argument and search against a database for homologous sequence segments, assigning an E-value to the results. By using a straight homology-based method, it is possible to search the *C. elegans* open reading frames against a broad set of protein sequences. Then any match against a known kinase, meeting some threshold criterion, would positively identify a potential *C. elegans* kinase. This turns out to be a viable strategy, although as a general-purpose algorithm it does not benefit in any way from the family-specific nature of the problem of finding a particular known protein domain.

Enter the pattern search strategies, including PSI-BLAST [Altschul et al., 1997] and Hidden Markov Models (HMM) [Eddy, 1998]. PSI-BLAST is a hybrid strategy that begins with a general-purpose search, but iteratively constructs a problem-specific pattern. HMM is more directly family specific, requiring clear *a priori* knowledge of a family or of some of its members. In the case of protein kinases, such *a priori* knowledge is readily available. Conceptually, the most natural choice for a pattern search algorithm was a Hidden Markov Model. Such a model "describes a family of proteins by assigning large probabilities to sequences in that family" [Krogh et al., 1994]. The description consists, essentially, of a vector of probabilities for each amino acid position, with the probability specifying the likelihood of a given amino acid occurring at that position. In order to construct such a model, considerable prior knowledge of a protein family is required, typically in the form of a set of representative sequences. Therefore, HMMs are not suitable for general database searching. Nor do they take into account higher-level correlations; such as those arising from structural features. These caveats aside, HMMs capture the family-specific nature of the problem of identifying protein kinases.

To construct a model of the kinase family, we started with sequences of kinase catalytic domains from yeast to human. To avoid biasing the model toward certain subfamilies, thus reducing sensitivity to other subfamilies of kinases, we effectively eliminated all kinases with homology in their catalytic domains greater than 50%. This left 70 broadly representative kinases. This general, 'representative' kinase profile served to identify all potential kinases in the *C. elegans* genome.

An HMM search may be used to identify complete or partial matches against the profile. It was crucial to search for partial matches, since short open reading frames could not be expected to contain a complete kinase. Rather, kinases might be spread across multiple exons, with frame-shifts in between. In fact, such was frequently the case. See Table I in which a cosmid denoted by its accession number ZC404, has five distinct exons in all three reverse strand reading frames. Some of these fragments have relatively low scores—low enough that they might easily be overlooked on their own. But combined, the five fragments yield a complete kinase domain with a highly significant score.

Using a custom gene assembly algorithm, all such fragments were assembled to form as many complete kinase domains as possible. The criteria were as follows: fragments must be on the same strand; they must consecutively span the kinase profile so that each additional fragment adds to the complete kinase domain; they must not overlap by 'too much', defined arbitrarily as a maximum of a 10 amino acid

overlap (results varied little in response to the precise cutoff value). No restrictions were placed on the length of introns, as intron lengths varied tremendously. After assembling kinase fragments, the assembled kinases were scored against the complete kinase HMM, this time forcing a global match. All fragments and assembled kinases were stored in the database, along with precise location and strand information, HMM scores and E-values, and the matching region in the kinase profile.

All analysis up through this point—from downloading the raw genomic sequence, storing it in a database, translating it an six frames, searching against an HMM profile, parsing the results of that search, storing the results in a database, assembling fragmentary kinases, and storing these in the database—was entirely automated using simple shell scripts and programs written in the Java language. It executes relatively quickly (on the order of a few hours using a standard single-processor desktop computer with average specifications), with the overwhelming majority of the time consumed in database access and HMM searching. With hardware-accelerated versions of the HMM algorithm just now becoming available, this portion of the analysis can be vastly sped up. By bypassing some of the intermediate database storage, or by accessing the database using a separate processor, the final bottleneck can be partly alleviated through a combination of hardware and careful optimization. Significantly, the entire automated pipeline scales linearly with respect to the length of the genomic sequence. By annotating sequence data incrementally, as it becomes available, the volume of data should remain manageable even as the larger *Drosophila* and human genomes become available.

With a scalable, fully automated pipeline in place for identifying *C. elegans* kinases, the automated portion of the annotation quickly reached its end. Human inspection and analysis were required to refine the initial output set down to a final set of 408 kinases, 21 kinase fragments and 82 kinase-like domains [Plowman et al., 1999]. Some redundant entries appeared, which were eliminated using pairwise Smith-Waterman analysis. Also, a few kinases spanned multiple cosmid clones and by examining the N- or C- termini of adjacent cosmids, it was possible to assemble the complete domains. Notably, the analysis of translated genomic data turned up approximately 40 kinase domains that were absent in the 19,099 protein data set separately available. This disparity suggests limitations in using gene prediction algorithms as the sole means of identifying coding regions. Our experience suggests that gene prediction algorithms should be supplemented by homology or profile search algorithms.

Finally, the putative kinases were clustered using hierarchical 'phylogenetic' methods. This required the creation of a multiple sequence alignment of all of the kinase catalytic domains which was created by aligning against the HMM. Then it was possible, using the Phylip package [Felsenstein, 1989], to create parsimonious tree structures approximating the relations among the kinases. At this point, the problem of data analysis became a problem of data visualization.

Vizualization

There were two primary types of data that needed to be visualized in the course of identifying potential protein kinases in *C. elegans*. As mentioned already, it was helpful to represent the various kinase families as a tree. Once a tree structure had been generated, we graphically viewed that tree in a variety of forms: an unrooted tree, a dendogram, and a less familiar format called a hyperbolic tree [Bingham and Sudarsanam, 1999a]. In addition to tree structures, other types of data needed visualization, specifically the HMM alignments and the position of putative kinases relative to their open reading frames raw sequences. A combination genome browser and sequence viewer served these purposes [Bingham and Sudarsanam, unpublished].

Visualizing a tree of over 400 nodes, one for each putative kinase, posed a variety of problems. It was difficult to search by eye through all the labels in order to find a particular kinase. It was hard to see in a glance whether the clustering met prior expectations, and also where exactly the clusters fell. Also, it was difficult to navigate the tree, zooming in on particular portions without losing track of the relationship between that portion of the tree and the rest of it. To address these needs, custom software was written to project large trees on to hyperbolic space [Bingham and Sudarsa-

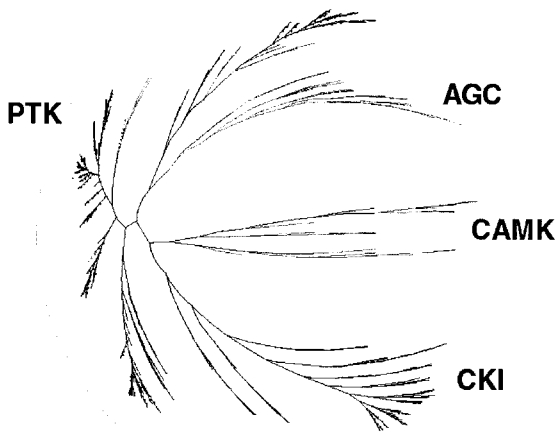

Fig. 2. Hyperbolic view of protein kinases of *C. elegans*.

nam, 2000]. A projection of *C. elegans* kinase tree in hyperbolic space is shown in Figure 2.

In addition to the tree viewer, a genome browser and sequence viewer was created. It can access SuCeDb or the local file system, and provides three distinct views of sequence data. The first view is a simple tabular view of accession number, locations, HMM scores, and so on. The second view shows pairwise or multiple sequence alignments, with color-coded amino acids and nucleic acids. Mismatches can be highlighted by color inversion. The third view represents a sequence, its open reading frames, and any feature predictions such as kinases, as horizontal bars. Base locations span the x-axis, while the y-axis supports features and sequences of various categories (see Fig. 3). As with the tree viewer, data can be color coded for intuitive, at-a-glance classification, and individual sequences or features can be selected from a list. Also, the three views are closely integrated, with selections in one view highlighting the corresponding data in the other views. With both of these software projects, software design posed its own set of challenges for sequence analysis, automation, and scalability.

Software Design

If not planned properly, software development time can easily exceed data analysis time. To minimize the time spent in writing software, all code was written in the Java language, proven to have development and debugging time less than half that of C or C++. Secondly, the programs were written using component-based design for maximal code reuse between applications and for the highest level of software maintainability. Individual components can be updated or interchanged without affecting the integrity of the complete system. Also, components can be quickly adapted to new purposes; for example, the alignment viewer can be used without modification to view HMM alignments, BLAST alignments, or multiple sequence alignments. By using current software development paradigms, tremendous time and energy has been saved, as great as any in the remainder of the analysis pipeline. Finally, it is worth noting that the performance-critical aspects of the visualization tools all scale linearly with respect to the number of sequences. A standard personal computer with typical RAM and clock speeds can support all visualization needs for the *C. elegans* kinases.

CONCLUSION

Many of the prerequisites for large-scale genomic sequence analysis are readily available. With the complete *Drosophila* and human genomes on the way, there can be no doubt about the 'large scale'—the pure magnitude—of the challenge that confronts genomics and bioinformatics. Fortunately, a range of algorithms and technologies already exist for dealing with this data, including relational databases, homology and profile search algorithms, scripting languages, and data visualization tools. While certainly partial and imperfect, they nonetheless provide a starting point. The challenge of large-scale sequence analysis is to carefully integrate the existing technologies, to build upon them where necessary, and to preserve data integrity and quality of analysis in the process.

With the completion of the *C. elegans* genome, the first fully sequenced multicellular organism, all of the problems and pitfalls of large-scale sequence analysis are becoming apparent. From our study, the following guiding principles have emerged: write scalable software, since more hardware may not be the solution; encapsulate domain-specific knowledge in the database design; use existing software when available, and write reusable component-based software when it is unavail-

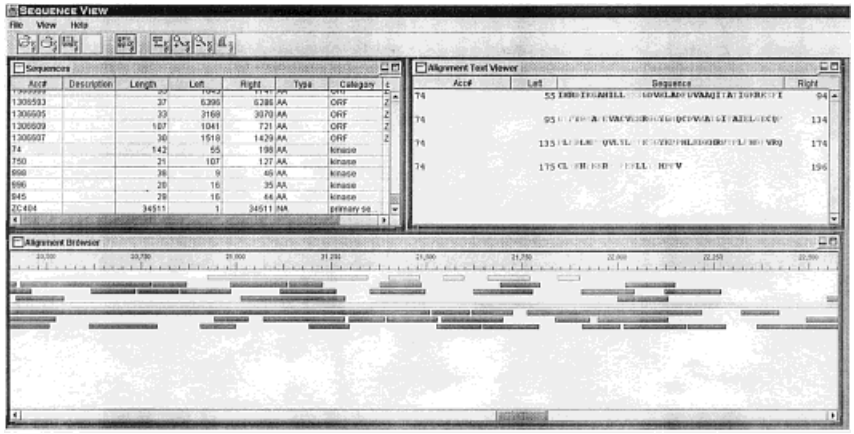

Fig. 3. Genome browser. Table in top left shows kinase exon predictions along with coordinates of these exons on a cosmid. Top right panel shows a color-coded view of the assembled kinase domain. Bottom panel shows the alignment of kinase exons (top row) on the cosmid (middle row). ORFs in six frames are shown above and below the cosmid.

able; automate analysis to the greatest extent possible. Finally, since there is no real substitute for a human expert in assessing the quality of sequence alignments, all tools should cater to human experts. By following these principles, the analysis pipeline outlined for protein kinases can scale up as larger complete genomes become available.

REFERENCES

Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25:3389–3402.

Bingham J, Sudarsanam S. 2000. Visualizing large hierarchical clusters in hyperbolic space. Bioinformatics. In press.

Eddy SR. 1996. Hidden Markov models. Curr Opin Struct Biol 6:361–365.

Felsenstein J. 1989. PHYLIP—Phylogeny Inference Package (Version 3.2). Cladistics 5:164–166

Henikoff JG, Henikoff S, Pietrokovski S. 1999. New features of the Blocks Database servers. Nucl Acids Res 27:226–228.

Krogh A, Brown M, Mian IS, Sjolander K, Haussler D. 1994. Hidden Markov models in computational biology: applications to protein modeling. J Mol Biol 235:1501–1531.

Plowman GD, Sudarsanam S, Bingham J, Whyte D, Hunter T. 1999. The protein kinases of *C. elegans:* a model for signal transduction in multicellular organisms. PNAS. 96:13603–13610.

Smith T, Waterman M. 1981. Identification of common molecular subsequences. J Mol Biol 147:195–197.

The *C. elegans* Sequencing Consortium. 1998. Genome sequence of the nematode *C. elegans*: a platform for investigating biology. Science 282:2012–2018.