PROTEIN SEQUENCE AND STRUCTURE COMPARISON ON MASSIVELY PARALLEL COMPUTERS

Robert Jones

THINKING MACHINES CORPORATION
CAMBRIDGE, MASSACHUSETTS 02142

Summary

Sequence comparison has become a standard tool in the analysis of newly determined protein sequences. As the database of known sequence grows not only does the cost of database searching increase but so too does the demand for that service. These factors conflict directly with the desire to use the most sensitive methods available. The use of massively parallel computers for database searching provides a solution to this problem and is helping in the development of new methods for both sequence and structure comparison.

The International Journal of Supercomputer Applications, Volume 6, No. 2, Summer 1992, pp. 138–146

Introduction

At the heart of computational molecular biology lie methods for comparing macromolecules. By comparing the sequences of regions of DNA one can locate potential genes and their regulatory signals. By comparing the sequences of proteins, the function of a novel sequence can be suggested and domains that are important for structure and function can be discovered. The comparison of the three-dimensional structures of proteins can help identify common folding domains and may lead to improved methods of structure prediction.

The power of these comparative methods is a direct result of the actions of natural selection. All the proteins that we observe today have evolved from ancestral forms, with natural selection favoring those most suited for the present environment. In the regions of a protein that perform a valuable function, evolution is tightly constrained with respect to the mutations that are acceptable. Less important parts of the molecule are less strongly constrained, and related proteins are likely to exhibit diversity in these regions. Proteins that perform the same function but are from distantly related species often contain clearly delimited domains of conserved sequence that are a strong indication of functional importance.

A striking example of the power of the conservation of sequence domains between distantly related proteins, and of the power of sequence comparison in establishing such relationships, involves the gene involved in the disease cystic fibrosis. At the time of its isolation the function of this gene was unknown. After the DNA sequence was determined, the translated protein sequence was compared with the database of all known proteins. Striking similarities were found to a large and diverse family of proteins involved in solute transport across cell membranes (Riordan et al., 1989). Figure 1 shows part of the alignment of the protein sequence with three bacterial proteins. On the basis of sequence similarity it was proposed that the human protein was involved in membrane transport and contained a binding site for ATP. Further work has revealed that the protein is involved in chloride ion transport, and that

knowledge is being used to develop therapies with which to treat the disease.

Sequence Comparison

Figure 1 also serves to illustrate one of the difficulties of sequence comparison in that the task is not simply one of string matching. Amino acids resemble each other to varying degrees according to the physical and chemical properties of their side chains. Based on some similarity metric that includes this information, sequence comparison algorithms must find the best alignment between two sequences, matching up positions to achieve the best alignment score and introducing gaps into the sequences if this would improve the score. This task can be described as a problem in optimization, for which the most sensitive method available is based on the technique of dynamic programming.

The most versatile algorithm that has emerged in this area is that of Smith and Waterman (1981) which identifies the most conserved segments within a pair of sequences. If the two sequences are closely related, then these regions are likely to cover the entire length of the molecules. If they are only distantly related, then the algorithm will identify the best region within otherwise very different sequences. When one wants to find distant relationships between proteins it is exactly this behavior that is desired. There are many variants of the basic method that identify the best N suboptimal alignments and produce alignments of multiple sequences (Taylor, 1987; Waterman and Eggert, 1987).

For practical purposes the sensitivity and versatility of dynamic programming must be viewed in the context of its relatively high computational cost. The comparison of two sequences can be accomplished by any workstation, but the thorough study of a sequence requires that it be compared with all other known sequences. Release 67 of the GenBank DNA sequence database contains 43,903 sequences, representing 55,169,276 nucleotides. Even with fast serial machines a database search using dynamic programming will take several hours. This problem is destined to get worse, as the size of the database is growing exponentially, as shown in Figure 2. With the initiation of projects to map

```
CTFR   FSLLGTPVLKDINFKIERGQLLAVAGSTGAGKTSLLMMIMG
               * * * *       *  *** *********  *
hisP   RRYGGHEVLKGVSLQARAGDVISIIGSSGSGKSTFLRCINF
pstB   FYYGKFHALKNINLDTAKNQVTAFIGPSGCGKSTLLRTFNK
malK   KAWGEVVVSKDINIDIHEGEFVVFVGPSGCGKSTLLRMIAG
```

Fig. 1. Sequence conservation between the protein involved in cystic fibrosis and three bacterial transport proteins By convention the 20 possible amino acids are represented as characters of the alphabet. Amino acids resemble each other to varying degrees as a result of the physical and chemical properties of their side chains. The most conserved positions in the alignment are shown in boldface.

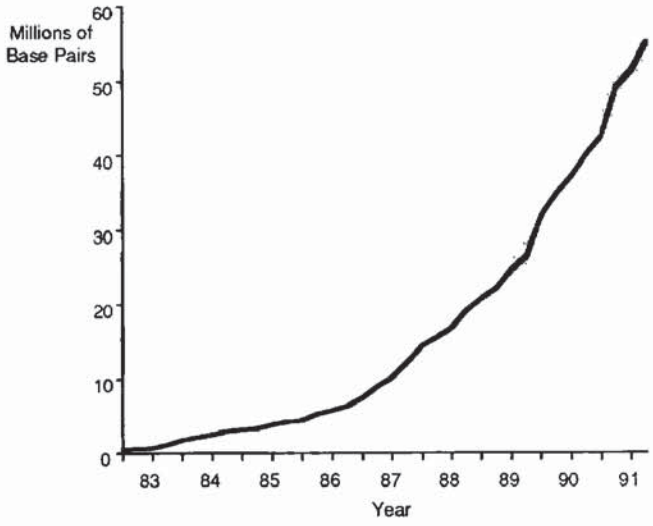

Fig. 2. Growth of the GenBank DNA sequence database

and sequence a number of genomes, most notably the Human Genome Project, this pattern of growth is likely to be sustained.

Several approaches have been followed to improve performance while attempting to retain the sensitivity of the original method. Software based on the search technique of hashing (FASTP) is widely used, and recently a faster method that employs a discrete state finite automaton search (BLAST) has been developed (Pearson and Lipman, 1988; Altschul et al., 1990). Both approaches have contributed greatly to the field but fail to achieve the best possible sensitivity and may miss important similarities between distantly related proteins.

Very high performance may be possible with custom VLSI devices that embody a specific algorithm, such as those commonly found in graphics workstations. Special-purpose hardware for sequence comparison has been designed and is under active development (Lipton and Lopresti, 1985; T. Hunkapiller, M. S. Waterman, R. Jones, J. Peterson, and E. Chow, unpublished results). Such devices are inevitably less flexible, in terms of the underlying algorithm, than a piece of software that may be changed at will to reflect improvements in the method. Here is where general-purpose supercomputers can play a major role, achieving high performance but retaining the flexibility of software.

Sequence Comparison on Massively Parallel Computers

As described above, the method of choice for sequence comparison is based on dynamic programming (Smith and Waterman, 1981). In this algorithm, for two sequences of length M and N, a matrix of M-by-N elements must be computed using the recurrence relation shown in Figure 3. The term $s(a_i,b_j)$ is the similarity between amino acid i of sequence a and amino acid j of sequence b. The term w is the cost of inserting a gap into one or the other of the sequences. The recurrence relation lays down the order in which elements of the matrix must be computed, in that the values for elements $(i-1, j-1)$, $(i, j-1)$, and $(i-1, j)$ must already be known before element (i,j) can be evaluated. The score of the best subalignment is simply the maximum score $H_{i,j}$ over the entire matrix (Fig. 4). The alignment itself is found by tracing back from the location of that maximum until the score falls to zero.

On massively parallel architectures the dependency that is inherent in the recurrence relation can be exploited to allow many elements to be computed in parallel. As shown in Figure 5, all elements that lie on an antidiagonal can be computed in parallel provided all those that lie above and to the left have already been computed. This is exploited in the implementation of this algorithm on the Connection Machine CM-2, a massively parallel computer with a SIMD architecture (Thinking Machines Corporation, 1991). In this particular application the processors of the machine are configured as a one-dimensional array that computes an entire antidiagonal at a time. To compute the entire matrix of M-by-N elements an array of M processors must compute $M + N - 1$ consecutive antidiagonals. In the context of the matrix one can visualize the array of processors sweeping across the matrix from left to right. Each processor is responsible for computing one row of the matrix, as shown in Figure 6 (Lander, Mesirov, and Taylor, 1988; Jones et al., 1990).

An alternative implementation of the algorithm on the AMT Distributed Array Processor (DAP) computes one row of the matrix in parallel, as opposed to an antidiagonal (Collins, Coulson, and Lyall, 1987; Collins and Reddaway, 1990). This has twice the processor utilization of the antidiagonal form and therefore achieves higher performance. Its disadvantage lies in its requirement for a restricted form of gap penalty, in which a single cost is charged for a gap regardless of length. The antidiagonal form allows the cost to rise as the length of the gap increases. In particular, it permits the most general case of affine gap penalties that appear to mimic most accurately the distribution of gaps in well-studied alignments.

On a 65,536 processor Connection Machine CM-2, the "antidiagonal" implementation of the algorithm can compute approximately 100 million matrix entries per second whereas the "row" implementation can achieve approximately 150 million matrix entries per second. The difference in processor utilization between the two approaches and the type of communications operations involved account for the performance figures. Nicholas

and associates (1991) have used a Connection Machine CM-2 to compute the scoring matrices and a CRAY Y-MP, linked to the CM-2 via a high bandwidth communications channel, to perform the tracebacks and alignments.

A very different approach is possible with coarse-grain parallel architectures such as the Intel iPSC/860 and the Connection Machine CM-5. Each node has a considerable amount of local memory and can run a totally separate program from what its neighbors are running. This has been exploited for database searching by simply placing a different section of the database in the memory of every node. A separate comparison is performed in each node, with the results being merged on completion of the task. This approach involves little change to existing serial codes and hardly any interprocessor communications. It has yielded performance of 12 million matrix entries per second on a 32-node Intel iPSC/860 (Deshpande, Richards, and Pearson, 1991) and 33 million matrix entries per second on a 32-node CM-5 (R. Jones, unpublished results).

Sequence Comparison with Multiple Alignments

There are numerous examples in which biologists have demonstrated a close structural or functional relationship between two proteins but the sequence comparison, as described above, has failed to detect any significant similarities. With further biological knowledge about the proteins involved, one might be able to bias the alignment process in some way such that the similarity was revealed. In the general case, however, no such information is available.

Some information of this form is generated automatically whenever two related protein sequences are aligned. From examination of the alignment it is clear that some positions are conserved and some differ. Given that the proteins have been subject to evolution, it is reasonable to assume that the conserved positions are more important to structure and function than the variable ones. These positions should receive a higher weight in the alignment process. Multiple sequence alignments may be used in comparisons with individual sequences using a method known as "profile analysis"

$$H_{i,j} = \max \begin{cases} 0 \\ H_{i-1,j-1} + s(a_i, b_j) \\ H_{i-1,j} + w \\ H_{i,j-1} + w \end{cases}$$

Fig. 3. The recurrence relation used in the Smith-Waterman algorithm $H(i,j)$ is the score in element (i,j), $s(a,b)$ is the similarity between two amino acids, and w is the cost of introducing a gap into the alignment.

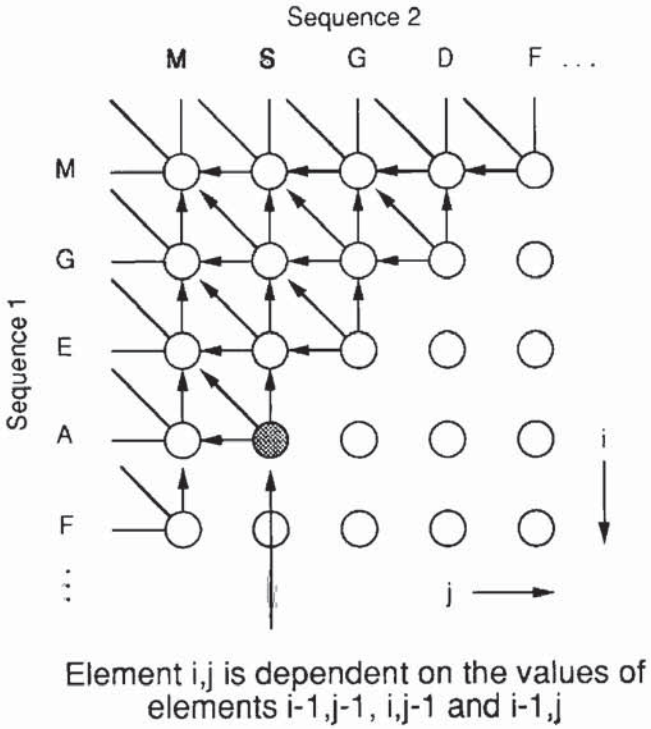

Fig. 4. Calculation of the scoring matrix

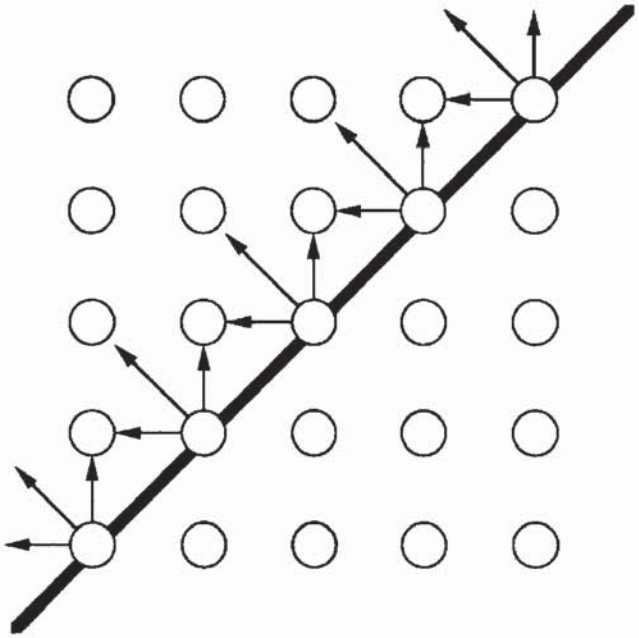

Fig. 5. All elements of a given antidiagonal can be computed in parallel

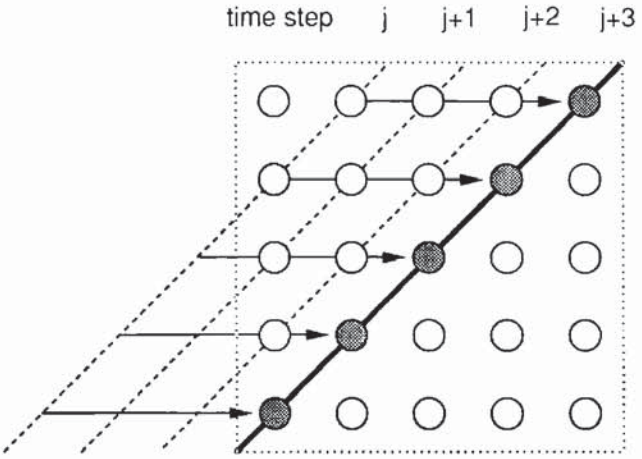

Fig. 6. On each iteration an entire antidiagonal is computed in parallel

(Gribskov, MacLachlan, and Eisenberg, 1987). Implicit in this approach is that conserved positions receive a higher weight and play a greater role in the alignment process.

The sequence comparison software on the CM-2 has been adapted to use alignments in this manner. Eric Lander and the author have used this approach to repeatedly search the sequence databases using a multiple sequence alignment (R. Jones, E. Lander, unpublished results). On every cycle the best matching database sequence is added to the multiple alignment. This approach shows promise in extending the sensitivity of database searches in those cases where an initial alignment of at least two sequences can be established. Because it involves repeated database searches this approach is not feasible for serial computers. This highlights the role of supercomputers in advancing not only the performance of existing software but also the sensitivity of these methods through more extensive computation.

Protein Structure Comparison

The determination of the full three-dimensional structure of a protein at atomic resolution provides biologists with a wealth of information. Such detailed knowledge, however, requires a great deal of work, and as a result there are approximately 600 structures known compared to approximately 30,000 protein sequences. Even with dramatic technological advances this disparity is not likely to change much in the near future. The ability to predict structure directly from sequence has therefore become a major research goal for many groups, and the success of this effort would revolutionize the field of molecular biology. The basic approach of structure prediction is to examine known structures, identify conserved structural motifs, and attempt to find sequence patterns that correlate with them. The comparison of three-dimensional structures is fundamental to these efforts, but until recently such comparisons have been performed using simple techniques that lack sensitivity and are not suited for database searching.

Most of these methods rely on the rotation and translation of one structure to minimize the root mean square difference between atomic coordinates (Mat-

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.