Journal of Computer Science 4 (1): 36-41, 2008
ISSN 1549-3636

Nimble Protein Sequence Alignment in Grid (NPSAG)

K. Somasundaram and S. Radhakrishnan
Department of Computer Science and Engineering,
Arulmigu Kalasalingam College of Engineering, Krishnankoil-626190, Tamilnadu, India

Abstract: In Bio-Informatics application, the analysis of protein sequence is a kind of computation driven science which has rapidly and quickly growing biological data. Also databases used in these applications are heterogeneous in nature and alignment of protein sequence using physical techniques is expensive, slow and results are not always guaranteed/accurate. So this application requires cross-platform, cost-effective and more computing power algorithm for sequence matching and searching a sequence in database. Grid is one of the most emerging technologies of cost effective computing paradigm for large class of data and compute intensive application which enables large-scale aggregation and sharing of computational data and other resources across institutional boundaries. We proposed the Grid architecture for searching of distributed, heterogeneous genomic databases which contained protein sequences to speed up the analysis of large scale sequence data and performed sequence alignment for residues match.

Key words: Grid computing, bio-informatics, sequence alignment, dynamic programming

INTRODUCTION

Grid computing, most simply stated, is distributed computing taken to the next evolutionary level. The goal is to create the illusion of a simple yet large and powerful self managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources.

Another key technology in the development of grid networks is the set of middleware applications that allows resources to communicate across organizations using a wide variety of hardware and operating systems. The promise of grid computing is to provide vast computing resources for computing problems that require supercomputer type resources in a more affordable way. Grid computing also offers interesting opportunities for firms to tackle tough computing tasks like financial modeling without incurring high cost for supercomputing resources.

Grid computing is applying the resources of many computers in a network to a single problem at the same time usually to a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. Grid computing is thought of as a form of network-distributed parallel processing. It can be confined to the network of computer workstations within a corporation or it can be a public collaboration.

Inexpensive systems such as Beowulf clusters have become increasingly popular in both the commercial and academic sectors of the bioinformatics community. Clusters typically consist of a master node that distributes the bioinformatics application amongst the other nodes. The PC clusters can be used to replace mainframe systems or supercomputers and save much hardware cost. According to efficiency and cost, using parallel version software and cluster system is a good way and it will become more and more popular in the near future. Grid computing offers significant enhancements to the capabilities for computation, information processing and collaboration.

Bioinformatics and computational molecular biology are concerned with the use of computing and mathematical sciences as tools to advance traditional laboratory based biology. The need to process an exponentially growing amount of biological information for further scientific advances and to understand its role in heredity, chemical processes within the cell, drug discovery, evolutionary studies etc.

Proteins are polymers also called polypeptides consisting of a sequence of amino acids. There are twenty amino acids that are found in proteins. Figure 1 shows the full name and abbreviation of 20 amino acid of protein.

Corresponding Author: K. Somasundaram, Department of Computer Science and Engineering,
Arulmigu Kalasalingam College of Engineering, Krishnankoil-626190, Tamilnadu, India
Tel: +91-4563-289042 Mobile: +91-9443467264 Fax: +91-4563-289322

Full Name	Abbrev.		Full Name	Abbrev.	
Alanine	A	Ala	Methionine	M	Met
Cysteine	C	Cys	Asparagine	N	Asn
Aspartic acid	D	ASP	Proline	P	Pro
Glutamic acid	E	Glu	Glutamine	Q	Gln
Phenylalanine	F	Phe	Arginine	R	Arg
Glycine	G	Gly	Serine	S	Ser
Histidine	H	His	Threonine	T	Thr
Isoleucine	I	Ile	Valine	V	Val
Lysine	K	Lys	Tryptophan	W	Trp
Leucine	L	Leu	Tryosine	Y	Tyr

Fig. 1: Protein names

Proteins were first characterized by their primary sequences, the amino acid sequence[1] and then folded into complex tertiary (3D) structure, which decided the corresponding biological functions. The motivation behind the structural determination of proteins was based on the belief that structural information would ultimately result in a better understanding of intricate biological processes.

Protein sequence alignment is one of the bioinformatics research projects, facilitating everything from identification of gene function to structure prediction of proteins. Alignment of two sequences showed how similar the two sequences were, where there were differences between them and the correspondence between similar subsequences. Similarity simply means that two sequences are similar, by some criterion. All of this represents important information for biologists. The successful techniques for prediction of the protein three dimensional structures rely on aligning the sequence of a protein of unknown structure. To attempt to align the protein sequence for large proteins, we needed better algorithms and larger computational resources like those afforded by either powerful super computer or distributed computing.

RELATED WORKS

The NdPASA[6] is a novel protein sequence pairwise alignment algorithm. This method employs neighbor-dependent propensities of amino acids as a unique parameter for alignment. NdPASA optimizes alignment by evaluating the likelihood of a residue pair in the query sequence matching against a corresponding residue pair in the template sequence. Statistical analysis of the performance of NdPASA indicated that the introduction of sequence patterns of secondary structure derived from neighbor-dependent sequence analysis clearly improved alignment performance for sequence pairs sharing less than 20% sequence identity. For sequence of pairs sharing 13-21% sequence identity NdPASA improved the accuracy of alignment over the conventional global alignment algorithm using BLOSUM 62 by an average of 8.6%

Pattern Hunter[7] is a general purpose homology search tool, it uses novel approaches to substantially improve sensitivity and speed simultaneously. One new idea in Pattern Hunter was the introduction of an optimized spaced seed. In Blast, exact matches of k continuous letters is used as a seed to find long matches around it, whereas in Pattern Hunter, a seed is k discontinuous letter matches, where the relative positions of the k letters are optimized in advance. This has helped Pattern Hunter to significantly increase its sensitivity over Blast. Given k seeds, computing the hit probability under the uniform distribution is NP-hard. The problem of finding k optimal seeds is NP-hard. Using optimized multiple spaced seeds; Pattern Hunter is faster than Smith-Waterman at approximately the same sensitivity, for DNA sequence search. But investigation is going on for new multiple optimal seed schemes to approximate the Smith-Waterman sensitivity for protein-protein searches.

In the subquadratic sequence alignment algorithm[8] data compression techniques were employed to speed up the alignment of two strings. Instead of dividing the dynamic programming matrix into uniform-sized blocks they employed a variable sized block partition and speeding up dynamic programming by keeping and computing only a relevant subset of important values. Here the dynamic programming solution to the string comparison computation problem can be represented in terms of a weighted alignment graph. The subquadratic sequence comparison algorithms presented were perhaps close to optimal in time complexity. However, an important concern was the space complexity of the algorithms. If only the similarity score value was required, the classical, quadratic time sequence alignment algorithm could easily be implemented to run in linear space by keeping only two rows of the dynamic programming table alive at each step. If the recovery of either global or local optimal alignment traces was required, quadratic-time and linear-space algorithms could be obtained by applying Hirschberg's refinement to the classical sequence alignment algorithms.

ParAlign[9] is a parallel sequence alignment algorithm specifically designed to take advantage of SIMD technology. The initial filtering method used in the ParAlign was very sensitive (few false negatives), but gave too many unwanted false positives in some cases. This happened occasionally with certain query sequences and was caused by repetitions in the sequences. An improved statistical evaluation method

was needed in order to improve performance. The Smith-Waterman algorithm was generally considered to be the most sensitive, but long computation times limited the use of this algorithm. Special purpose hardware with parallel processing capabilities performed smith-waterman searches at high speed, but these machines were expensive.

PROPOSED SYSTEM (NPSAG)

NPSAG has three grid sites named as Site1, Site2 and SiteN were connected to the grid environment as shown in Fig. 2. Each site had more than one grid node. Grid Index Information Server (GIIS), Global scheduler (GS), Local Scheduler (LS), Sequence Alignmenter (ALIG) and Sequence Updater were the components of NPSAG. User can get the services available in a Grid using GIIS and submit the sequence alignment of some protein structure as a request to GS through Grid GUI. Global Scheduler (GS) will direct the jobs to the local grids and execute the tasks in local grids using Local Scheduler (LS).

In each grid, the service discovery discovered what were the services and grid nodes available and collected the information from local sites and updated the same to GIIS. Each grid node became a peer node. All nodes in the grid had equal capability.

User can login to Grid using Grid GUI and search similar sequences for the particular protein sequence. NPSAG searched the grid and found out the more suitable protein sequence from GIIS and distributed to the destinations grid sites through the use of Global Scheduler. Once the location is found out from the NPSAG, direct communication will be established to the desired grid sites and align the protein sequence for residues match using alignmenter. NPSAG has 2 main alignmenters namely Local Alignmenter (LALIG) and Global Alignmenter (GALIG).

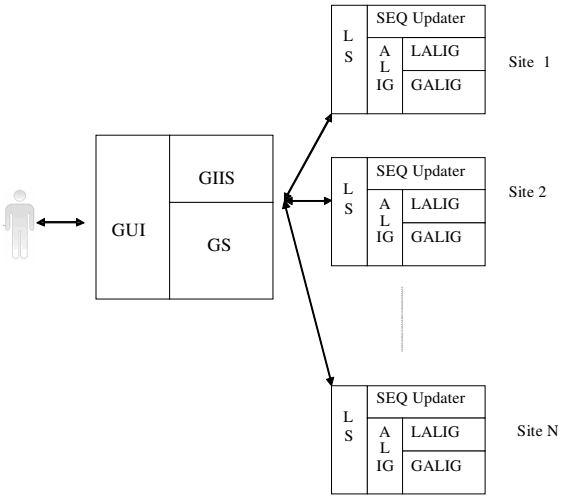

Fig. 2: Architecture of proposed system

If any new protein sequences are discovered by a person who is the participant of grid environment then he can update these details to the grid GIIS with the use of content distribution algorithm[5]. Content distribution system creates a distributed storage medium that allows for the publishing, searching and retrieval of files by members of its network. By use of content distribution the new data are updated to the GIIS in a faster manner. The Gird server in the NPSAG updates this information to in the GIIS. From the GIIS it can be distributed to the local grids with the use of sequence updater.

IMPLEMENTATION

To form an alignment between two sequences, spaces were inserted in arbitrary positions in the sequences so that they ended up with same length and then each character or space in one sequence would have a corresponding character or space in the other sequence. An alignment score can then be assigned to such an alignment: if a character is in sequence A matches its corresponding character in sequence B, it will receive a score of 1 (match); otherwise it will receive a score of -1(mismatch) and if one of the two characters is a space, it will receive a score of -2 (gap) and the total score over the whole sequence is the score of this alignment. The optimal alignment problem was to find the maximal score of all possible alignments between two sequences. This maximal score can be used to measure the similarity between the two sequences.

Computational approach for sequence alignment generally falls into two categories: global alignment and local alignment. Global alignment is a form of alignment that assumes that the two proteins are basically similar over the entire length of one another. By contrast, a local alignment searches for segments of the two sequences that match well. There is no attempt to force entire sequences into an alignment; just those parts that appear to have good similarity, according to some criterion identify regions of similarity within long sequences that are often widely divergent overall. Global alignments, which attempt to align every residue in every sequence, are most useful when the sequences in the query set are similar and of roughly equal size. A general global alignment technique is called the Needleman- Wunsch algorithm and is based on dynamic programming. Local alignments are most useful for dissimilar sequences that are suspected to

contain regions of similarity or similar sequence motifs within their larger sequence context. The Smith-Waterman is general local alignment method also based on dynamic programming.

Global Alignment:

F T F T A L I L L A V A V
F - - T A L - L L A - A V

Local Alignment:

F T F T A L I L L - A V A V
- - F T A L - L L A A V - -

Global and Local Alignments

GALIG: The standard global alignment algorithm computes the similarity between two sequences A and B of lengths m and n, respectively, using a dynamic programming approach. Dynamic programming is a strategy of building a solution gradually using simple recurrence. The key observation for the alignment problem was that the similarity between sequences A [1...n] and B [1...m] could be computed by taking the maximum of the three following values:

- The similarity of A [1...n -1] and B [1...m -1] plus the score of substituting A[n] for B[m]
- The similarity of A [1...n -1] and B [1...m] plus the score of deleting aligning A[n]
- The similarity of A [1...n] and B [1...m -1] plus the score of inserting B[m]

From this observation, the following recurrence can be derived:

sim (A[1..i], B[1..j]) = max {
sim (A[1..i -1], B[1..j -1]) + sub (A[i], B[j]);
sim (A[1..i -1], B[1..j]) + del (A[i]);
sim (A[1..i], B[1..j -1]) + ins (B[j]) }

Where sim (A, B) is a function that gives the similarity of two sequences A and B and sub (a, b), del (c) and ins (c) are scoring functions that give the score of a substitution of character a for character b, a deletion of character c and an insertion of character c, respectively. This recurrence is complete with the following base case: sim (A [0], B [0]) = 0

Where A[0] and B[0] are defined as empty strings. To solve the problem with this recurrence, the algorithm builds an (n +1) × (m +1) matrix M where each M[i, j] represents the similarity between sequences A[1..i] and B[1..j] The first row and the first column represent alignments of one sequence with spaces. M [0, 0] represents the alignment of two empty strings and is set to zero. All other entries are computed with the following formula:

M[i, j] = max {
M[i -1, j -1] + sub (A[i], B[j]);
M[i -1, j] + del (A[i]);
M [i, j -1] + ins (B[j]) }

The matrix can be computed either row by row (left to right) or column by column (top to bottom). In the end, M [n, m] will contain the similarity score of the two sequences. Once the matrix has been computed, the actual alignment can be retrieved by tracing a path in the matrix from the last position to the first. The trace is a simple procedure that compares the value at each M [i, j] to the values of its left, top and diagonal entries according to the formula given above. It is often useful to see the dynamic programming solution for the sequence alignment problem as a directed weighted graph with (n +1) × (m +1) nodes representing each entry (i, j) of the matrix and having the following edges:

- ((i -1, j -1), (i, j)) with weight equals to sub (A[i], B[j])
- ((i -1, j), (i, j)) with weight equals to del (A[i])
- ((i, j -1), (i, j)) with weight equals to ins (B[j])

A path from node (0, 0) to (n, m) in the alignment graph corresponds to an alignment between the two sequences and the problem of retrieving an optimal alignment was converted to the problem of finding a path in the graph with highest weight.

LALIG: Local alignment was defined as the problem of finding the best alignment between substrings of both sequences. The main difference was that M[i, j] contains the similarity between suffixes of A[1..i] and B[1..j]. As a result, the recurrence relation is slightly altered because an empty string is a suffix of any sequence and, therefore, a score of zero is always possible. The formula for computing M [i, j] becomes:

M [i, j] = max {0;
M[i -1, j -1] + sub (A[i], B[j]);
M[i -1, j] + del (A[i]);
M[i, j -1] + ins (B[j]) }

Another important distinction was that the score of the best local alignment was the highest value found anywhere in the matrix. This position was the starting

point for retrieving an optimal alignment using the same procedure described for the global alignment case. The path ended, however, as soon as an entry with score zero was reached. It is trivial to see that the Smith-Waterman algorithm has the same time and space complexity as the Needleman-Wunsch[10].

The dynamic programming method is guaranteed to find an optimal alignment given a particular scoring function; however, identifying a good scoring function is often an empirical rather than a theoretical matter. Although dynamic programming is extensible for more than two sequences, it is prohibitively slow for large numbers or extremely long sequences. This method requires large amounts of computing power or a system whose architecture is specialized for dynamic programming. Hence the computation complexity of this problem can be overcome by using a dynamic hierarchical environment like Grid Computing.

PERFORMANCE ANALYSIS

Here we have included a sample observation of our work to indicate the behavior of protein sequence alignment, the graphs in Fig. 3 and 4 indicates that the time and number of proteins are directly proportional to each other as the number of protein for analysis increases, the time required to analyze also increases.

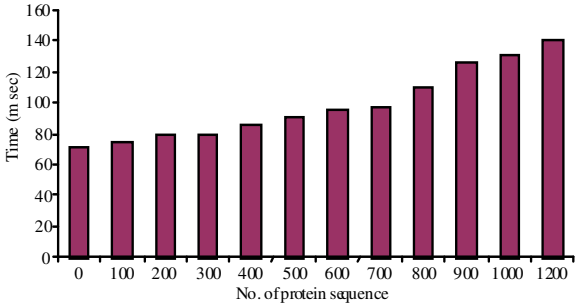

Fig. 3: Performance analysis in single system

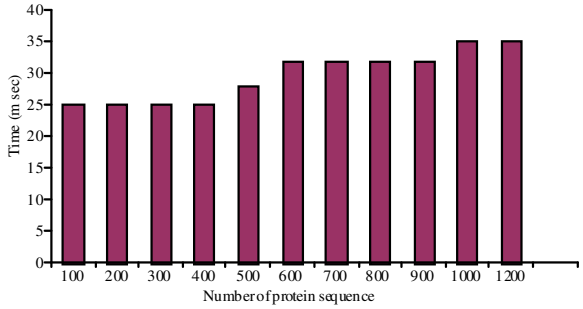

Fig. 4: Performance analyses in grid environment

Better accuracy was achieved when we performed analysis over a large database of proteins and hence the degree of accuracy improved over the increase of proteins.

To cope with the computational requirements for analysis on a large database, our work included Grid Computing environment. In grid computing environment time for sequence alignment was reduced. Hence our approach was a nimble process.

CONCLUSION AND FUTURE WORK

The physical methods, X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy can accurately perform protein sequence alignment; but these experimental methods are labor intensive and time consuming and for some proteins are not applicable at all. The computational prediction of accurate protein structure from the amino acid sequence remains a big challenge. It requires large computing power like super computer.

We have proposed an approach in aligning the protein sequence using Grid Computing. Our prediction approach matches the given sequence with the genomic database in one grid and provides the similar sequence. This was performed the sequence alignment in local grid using the dynamic programming method. Content distribution algorithm is used to distribute the sequence information to all other grids. Our approach has several advantages. First, it provides an interesting measure, match rate, for any protein sequence. Second, future improvement of our approach is incremental: as more protein structures are discovered each month, the alignment accuracy will likely get better automatically. Another advantage is that, the architecture of our system is flexible enough to allow other biological knowledge as well as machine learning techniques to be incorporated into the model to further improve its alignment accuracy. Our system could make a very useful addition to the current armory of sequence alignment methods available to protein chemists, genome annotators and bioinformaticians.

In future, we plan to implement the same problem in semantic grid. In GALIG and LALIG, we use some efficient Genetic algorithm operators to improve the over all performance of NPSAG.

REFERENCES

1. Scott Montgomerie, Shan Sundararaj, Warren J. Gallin and David S. Wishart 2006. Improving the accuracy of protein secondary structure prediction using structural alignment. Proc. BMC. Bioinformatis.

2. Huzefa Rangwala and George Karypis, 2006. Incremental window-based protein sequence alignment algorithms. Proc. of Bioinformatics.
3. Nicholas R. Jennings, David De Roure and Nigel R. Shadbolt, 2005. The Semantic Grid: Past, Present and Future. Proc. of IEEE.
4. David De Roure, 2005. A Brief History of the Semantic Grid. Proc. of Dagstuhl Seminar.
5. Stephanos Androutsellis-Theotokis and Diomidis Spinellis, 2004. A Survey of Peer-to-Peer Content Distribution Technologies. Proc. ACM Computing Surveys, 36.
6. Junwen Wang and Jin-An Feng, 2004. NdPASA: A novel pairwise protein sequence alignment algorithm that incorporates neighbor-dependent amino acid propensities. Proteins.
7. Ming Li and Bin Ma, 2003. PatternHunterII: Highly sensitive and fast homology search. Proc. Genome Informatics.
8. Siam J. Comput, 2003. A Subquadratic sequence alignment algorithm for unrestricted scoring matrices. Proc. Soc. Ind. Applied Math.
9. Torbjorn Rognes, 2001. ParAlign: A parallel sequence alignment algorithm for rapid and sensitive database searches. Nucleic Acid Res., 29.
10. James A. Cuff and Geoffrey J. Barton, 1999. Evaluation and Improvement of Multiple Sequence Methods for Protein Secondary Structure Prediction. PROTEINS: Structure, Function and Genetics.