International Journal of Computer Applications (0975 - 8887)
Volume 50 - No. 22, July 2012

Protein Sequence Similarity Search Technique Suitable for Parallel Implementation

Himanshu S Mazumdar
Head, Research & development Center
Dharmsinh Desai University
Nadiad, Gujarat, India

Maulika S Patel
Research Scholar
Dharmsinh Desai University
Nadiad, Gujarat, India

ABSTRACT

Having entered the post genomic era, there lies a plethora of information, both genomic and proteomic. This provides quite a lot of resources so that the computational and machine learning strategies be applied to address the problems of biological relevance. Searching in biological databases for similar or homologous sequences is a fundamental step for many bioinformatics tasks. On discovery of a new protein sequence or drug, a biologist would like to confirm the discovery by comparing with the largest available protein database. Alignment based methods become too complex and time consuming with the increase in the number of sequences. Alignment free sequence comparison is many a time used as a filtering step for application of alignment. A novel method of searching for similar sequences in a huge protein database is proposed. The method has two interesting aspects. One is the divide and conquer approach and use of hashing like scheme for indexing the large database. The index consists of the addresses of the 15-residue words in the UniRef100.fasta database. The second aspect is the possibility of data parallelism as the database is divided into m segments for indexing. This can further increase the efficiency of the algorithm. The creation of index is time consuming but the search time is constant and affordable. The method is particularly useful when used with the large databases like UniRef100.fasta which consists of 9757328 protein sequences as on May 2010. The index based searching algorithm is implemented in C # .NET.

General Terms:

Protein sequence similarity, alignment free

Keywords:

15- residue words, proteins, indexing, divide and conquer

1. INTRODUCTION

The post-genomic era is experiencing genomic and proteomic data floods and encouraging more researchers to address problems like targeted drug discovery, protein-protein interaction identification, protein function identification and more. It is well understood that searching is an important step towards finding homology, gene identification, motif identification, and other bioinformatics tasks[3, 4, 5]. Biologists are interested in identifying which sequences in a database are the most similar to a new sequence which is uncharacterized[9]. Alignment based algorithms[10, 11] have been proposed, but they suffer from the curse of dimensionality. As the number of sequences to be aligned increases, the complexity increases[13]. Heuristic based alignement algorithms like BLAST[15] are also very popular for sequence alignment. In this scenario, alignment free algorithms[2] have attracted many researchers. Different metrics have been proposed to assess the similarity obtained using alignment free techniques[1, 7, 8, 16]. A presearch approach is proposed in [14] to search for similar sequences from a huge database. The method worked by discovering first similar sequence and then using the common words for discovering similar sequences. Efficient sequence similarity searching becomes even more more challenging when the size of the database is huge. Indexing or hashing can reduce the latency. In the same light, an indexed based divide and conquer algorithmic method has been proposed and implemented for retrieving similar sequences. The method is also suitable for parallel implementation which can further increase the efficiency of the method.

2. MATERIALS AND METHODS

As shown in Fig. 1, the first step is to extract all 15 residue words from the database and prepare an index containing the location of these words in terms of sequence number in the database. To identify the location of all 15 residue words in a database is an expensive task in terms of time and storing the location is expensive in terms of space, which is not of much concern. If done in the simplest way possible, the index will have a list of 20^{15} entries containing the sequence number in the database. The space requirement is further increased with the use of a larger database, which usually is the case in proteomic tasks. The algorithm is used and tested with UniRef100.fasta, a comprehensive and non-redundant UniProt reference cluster, and ss.txt, a FASTA formatted file with protein sequences and secondary structures, databases available at www.uniprot.org. The UniRef100.fasta database is 4.21 GB in size (9757328 sequences) as of May 2010[12] and ss.txt, a smaller dataset, contains 174372 sequences. It is obvious that this large database cannot be handled by any efficient run time environment. To prepare the index, a divide and conquer strategy is adopted. We chose to segment the database into m segments or parts, such that each of the m segments consisted of around 100000 words baring the last segment. This facilitated us to handle the database at run time. The index is so prepared so

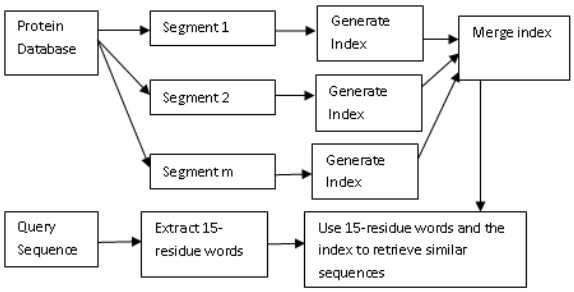

Fig. 1. Working of the indexed based search algorithm

as to point to those segments containing the considered word w in constant time. A linear search for the word w is carried out within the segment. The index helps us to quickly identify the segments of interest, where in a linear search is carried out. This clearly gives a speedup by a factor of m. The algorithm is m times faster as compared to linear search. The potential requirement is the creation of index which takes days of computing time on an Intel Core2Duo machine. The algorithm is described below:

Algorithm: Sequence similarity search
Input: Protein sequence database and query protein sequence
Output: Similar sequences

(1) Create index
(2) Use index
(3) Prune results

2.1 Creation of the index file:

The algorithmic steps for index creation are described below:

(1) Divide the database into m segments so as to conveniently handle the segments at run time. We have done this by creating a new segment if the number of words in the first segment exceeds 100,000 (limited by available memory).
(2) For each segment, do the following:
   (i) Prepare a list of 15 -residue words. These words are grouped into 400 files, based on the first two letters. That is they are hashed on the first two amino acid residues of the word.
   (ii) This is done by considering non duplicate 15-residue words occurring in the database having a frequency greater than one.
   (iii) Compute the frequency of the words and prepare files of type AA.txt.freq, AC.txt.freq, ..., YY.txt.freq. These files consist of the entries of the type, (word, frequency).
   (iv) An index, AA.indx, AC.indx, ..., YY.indx containing the word numbers from the files of type AA.txt.freq, AC.txt.freq, ..., YY.txt.freq, is created in each of the segments.
(3) A final index is prepared in the form of 400 files namely AA.indx, AC.indx, Ad.indx, ...,YY.indx, consisting of entries of the type, (word, segments).

The final index preparation process is the most time consuming of all as it combines the frequencies and locations of all words distributed over various segments. Finally, we have the direct address of each and every 15-residue word occuring more than once in the database. Words having single occurrence do not have any role to play in the similarity search process and hence are not allowed to participate in the indexing process.

2.2 Using the index

Once the index is created, do the following for the given query sequence:

(1) Generate a list of 15-residue words in the query sequence.
(2) Access the required files of the type AA.indx, AC.indx, ...YY.indx depending on the 15-residue word list.
(3) Identify the required segments, in which the words be looked for, based on the contents of the index files.
(4) Search the words in the respective segments sequentially for the containership of the words in the sequences.
(5) The sequences containing the word are appended to the list of similar sequences.

2.3 Prune the results

A sequence having at least one 15-residue word match with the query sequence will have an opportunity to be categorised as similar. However, based on the number of sequences retrieved, we have pruned the results to accomodate the most relevant similar hits.

3. RESULTS AND DISCUSSION

Searching in genomic and proteomic databases is challenging given the size of the databases. Linear search is not affordable and so index based methods are of particular interest. A similar method is implemented that builds an index of a large database on 15-residue words. The search becomes m times faster than the linear search where m is the number of segments in which the database is divided. 'm' is so chosen so as to handle the segments at run time while preparing the index. Table 1 shows the speedup for the two databases, Uniref100.fasta and ss.txt. It is assumed that finding a match of a long 15-residue word in two sequences cannot be by chance. The score of similarity increases with the increased number of common words between the query sequence and the target sequence. Database indexing is time consuming in our case. Also the databases are updated periodically. To incorporate the updates in the database, the master database and the current database are independently searched and the results are merged. The algorithm works by dividing the data into segments. The individual segments are independent of each other. Given the advancement in computer

Table 1. Speed Up Achieved.

Database	Number of Sequences	m
Uniref100.fasta	9757328	3292
SS.txt	174372	38

hardware and multiprocessing environment, the algorithm can be suitably adapted for data parallelism. This can further increase the efficiency of searching proportional to m.
The authors wished to compare the proposed method with the existing state of art methods. However, researchers use different approaches and it would not be fair to compare the two methods whose working is different. But through the result analysis, it is quite convincing that the method finds most of the similar sequences very fast. Speed is the salient feature of the proposed method. The weaknesses of the method include:

(1) Index preparation is time consuming.
(2) If not a single word of length 15 or greater is found, the sequence will not be considered as similar, though it may be similar in rare cases.

Following modifications are proposed to remove the weaknesses:

(1) Index preparation is one time and offline and hence is not of prime concern.
(2) By using the 15 residue words for search, sensitivity may be compromised. A secondary index on 14 residue words and less may be prepared and can used in case of requirement of increased sensitivity.

4. CONCLUSION

Sequence similarity searching is an important step in many bioinformatics tasks. The state of art methods are reviewed in brief and a method to extract the similar protein sequences from a large database is proposed. The method is particularly suitable for parallel implementation. This is required in the light of rapidly increasing data, both genomic and proteomic.

Acknowledgment

The authors would like to thank the Faculty of Pharmacy, DDU for discussion on targeted drug discovery requirements.

5. REFERENCES

[1] Tuan D. Pham and Johannes Zuegg. A probabilistic measure for alignment free sequence comparison. *Bioinformatics*, Advance Access:3455–3461, December 2004.

[2] Susana Vinga and Jonas Almeida. Alignment-free sequence comparison-a review.
Bioinformatics, 19:513–523, 2003.

[3] Nikola Kasabov. Bioinformatics: A Knowledge Engineering-Approach. *Second IEEE International Conference On Intelligent Systems*, June 2004.

[4] Achuthsankar S. Nair. *Computational Biology & Bioinformatics: A Gentle Overview.* Communications of the Computer Society of India, January 2007.

[5] C. Setubal and J. Meidanis. Introduction to Computational Molecular Biology, Cengage Learning, 1997

[6] J Chen and N. Chaudhari. Cascaded Bidirectional Recurrent Neural Networks for Protein Secondary Structure Prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics.* 4(4), Oct-Dec 2007.

[7] Weizhong Li, and Adam Godzik, Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22:1658–1659, Advance Access published on July 1, 2006

[8] Miriam R. Kantorovitz, Gene E. Robinson, and Saurabh Sinha, A statistical method for alignment-free comparison of regulatory sequences, Bioinformatics 23: Vol. 23 ISMB/ECCB 2007, pages i249–i255.

[9] Clare Sansom. Database searching with DNA and protein sequences: An introduction. Briefings in Bioinformatics (2000) Vol.1, No.1 (22–32).

[10] Saul B. Needleman and Christian D Wunsh. A general method applicable to thesearch for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology.* 48, 443–453, 1970

[11] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences, *Journal of Molecular Biology.* 147, 195–197, 1981.

[12] Baris E. Suzek, Hongzhan Huang, Peter McGarvey, Raja Mazumder, and Cathy H. Wu. UniRef: comprehensive and non-redundant UniProt reference clusters.*Bioinformatics*, 23:1282–1288, Advance Access published on May 15, 2007.

[13] Carsten Kemena and Cedric Notredame, Upcoming challenges for multiple sequence alignment methods in the high throughput era. *Bioinformatics* 2009.

[14] Maulika S Patel and Himanshu S Mazumdar. Similarity search using pre-search in UniRef100 database. *International Journal of Hybrid Information Technology.* 4(3), 31–40, July 2012.

[15] Altschul, S. F. et al. Basic Local Alignment Search Tool. *Journal of Molecular Biology.* 215, 403-410, 1990.

[16] Gesine Reinert, David Chew, Fengzhu Sun, and Michael S. Waterman,Alignment-Free Sequence Comparison (I): Statistics and Power, *Journal of Molecular Biology.* 16(12),1615-1634 December 2009.