Protein Structure Alignment Algorithms and Visualization Web Services with Various Initial Methods on Cloud Platform

Yaw-Ling Lin  Chen-En Hsieh  Guan-Jie Hua
Department of Computer Science and Information Engineering,
Providence University, Taichung, Taiwan
yllin@pu.edu.tw, {seanm7739,gt758215}@gmail.com

Che-Lun Hung
Department of Computer Science and Communication Engineering,
Providence University, Taichung, Taiwan
clhung@pu.edu.tw

Abstract

The biological function of a protein molecule is decided by its 3D-shape, which eventually determines how the molecule interacts with other molecules in living cells. Identifying similar structures between proteins provides the opportunity to recognize homology that is undetectable by sequence comparison. Thus comparison and alignment of protein structures represent a powerful means of discovering functions, yielding direct insight into the molecular mechanisms.

This paper proposes approaches in providing visualization tools for pairwise 3D protein structure alignment; our web service takes advantage of the MapReduce paradigm as means of management and parallelizing tools under massive number of protein pairs examined under the experiment. It shows that our previously proposed sequential combinatorial algorithms are well parallelized under the map/reduce platform. These methods are tested on the real-world data obtained from the RCSB PDB data set; the computation efficiency can be effectively improved proportional to the number of processors being used.

The VRML and Jmol are visualization tools to illustrate 3D protein structure, and we use the VRML to illustrate our protein structure comparison result.

Keywords: protein structures comparisons, bioinformatics, visualization, VRML, MapReduce, Hadoop, cloud computing.

1 Introduction

Protein structures play critical roles in vital biological functions [1]. To date, there are more than 83,000 protein structures determined by the advances in X-ray crystallography and NMR spectroscopy, molecular biologists these days proceed in the direction of analyzing and classifying these protein structures in order to discover the structural relationships with protein functions [2].

Detection of proteins with a similar fold can suggest a common ancestor, and often a similar function [3]. Comparison of 3D structures makes it possible to establish distant relationships, even between protein families distinct in terms of sequence comparison alone. This is why structural alignment of proteins increases our understanding of more distant evolutionary relationships [4, 5]. The link between structural classification and sequence families enables us to study functions of various folds, or whole proteins [6].

There have been several methods proposed to compare protein structures and measure the degree of structural similarity based on alignment of secondary structure elements as well as alignment of intra and inter-molecular atomic distances. The basic ideas are rapid identification of pair alignments of secondary structure elements, clustering them into groups, and scoring the best substructure alignment. For examples, the VAST system [7] is based on continuous distribution of domains in the fold space. The FSSP/DALI system [8] provides two levels of description - a coarsegrained one and one with a fine-grained resolution. The method, CATH, provides the complete PDB fold classification by domains and links to other sources of information. The two methods, CE and LGscore2 [9] focus on the local geometry rather than global features such as orientation of secondary structures and overall topology (as in the case of VAST or DALI). VAST has been used to compare all known PDB domains to each other. The results of this computation are

Note that there must be an atom-pairing scheme before one can do the structure alignment computation. The first atom of the first selection is compared to the first atom of the second selection, fifth to fifth, and so on. Incorporating ideas with bipartite matching and 3-parameter isometric transformation, Lin et al. [10, 11] proposed methods of using parametric searching strategies with adaptive controls, and demonstrated that more accurate and similar protein structure pairings are possible comparing to previous known results like VAST [7] or CE [9].

One of the crucial steps of these algorithms is finding a good isometric transformation, which leads to the best atom-pairing alignment between two proteins. In this paper, we propose algorithms for efficiently locating more suitable isometric transformations of one structure and aligning it to the other structure. Based upon the periodic property of the parametric settings, we propose parametric searching strategies by approximations with power series and trigonometric series. We show the effectiveness of the proposed parametric searching strategies by a set of experiments, which leads to better alignments of structure pairing in general.

Hadoop [12] is a software framework intended to support data-intensive distributed applications. It is able to process petabytes of data with thousands of nodes. Hadoop supports MapReduce programming model [13] for writing applications that process large data set in parallel on Cloud Computing environment. The advantage of MapReduce is that it allows for distributed computing of the map and reduction operations. Each map operation is independent of the others and all maps can perform the tasks in parallel. In practice, the number of the map is limited by the data source and/or the number of CPUs near that data. Similarly, a set of reducers can perform the reduce operations. All outputs of the map operations which share the same key are presented to the same reducer, at the same time. In addition, one of the important benefits to use Hadoop to develop the applications is due to its high degree of fault tolerance. Even when running jobs on a large cluster where individual nodes or network components may experience high rates of failure, Hadoop can guide jobs toward a successful completion. Many applications of bioinformatics are often computation consuming; sometimes it needs weeks or months to complete the jobs. The traditional parallel models, such as MPI, OpenMP and Multi-thread, are not suitable to such applications, where a fault occurred in some nodes leads the entire application into total failure. In these situations, the Hadoop platform are considered as a much better solution for these real-world applications. Recently, Hadoop has been applied in various domains in bioinformatics[14, 15].

2 Method and Materials

2.1 Protein structure Analysis

The main idea of our local refinement algorithm for finding a suitable matching between two sets of points before utilizing the RMSD procedure to fine-tune the final result is by adjusting the suitable parameter sets by ways of searching the underlying parametric space.

Root mean squared deviation

The smallest root mean squared deviation (rmse) is a least-squares fitting method for two sequences of points [8]. The idea is to align atom vectors of the two given (molecular) structures, and use the common least averaged squared errors as a measurement of differences between these two (paired) sequences. Formally, let \( P = \{ \mathbf{p}_1, \ldots, \mathbf{p}_n \} \) and \( Q = \{ \mathbf{q}_1, \ldots, \mathbf{q}_n \} \) be two sequences of points. We assume that \( P \) is translated so that its centroid \( \frac{1}{n} \sum_{i=1}^{n} \mathbf{p}_i \) is at the origin. We also assume that \( Q \) is translated in the same way. For each point or vector \( \mathbf{x} \), let \( \{ x_i \}_{i=1,2,3} \) denote the \( i \)-th \((x, y, z)\) coordinate value of \( \mathbf{x} \), and \( \| \mathbf{x} \| \) denote the length of \( \mathbf{x} \). Let

\[
\text{RMSD} (P, Q, R, a) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \| R \mathbf{q}_i + a - \mathbf{p}_i \|^2 },
\]

where " \( R \) " is a rotation matrix and " \( a \) " is a translation vector. Then, the \( \text{rmsd} \) value \( d(P, Q) \) between \( P \) and \( Q \) is defined by

\[
d(P, Q) = \min_{R, a} \text{RMSD} (P, Q, R, a),
\]

We refer to Martin's ProFit package (standing for protein fitting system) [16] and write a program to calculate the \( \text{rmsd} \) between C-a atoms of paired protein backbones with C language. Fitting was performed using the McLachlan algorithm [17].

Isometric rotation transformation

According to Euler's rotation theorem [18], any rotation about the origin point can be described by using three angular parameters. The rotation is determined by 3 consecutive rotations with 3 Euler angles \((\alpha, \beta, \gamma)\). The first rotation is done by the angle \( \alpha \) around the z-axis, the second is done by the
angle $\beta$ around the $x$-axis, and the third rotation is done by the angle $\gamma$ around the $z$-axis. Please refer to [19] for more detailed discussions about the transformation.

Analog to Euler’s setting, we can also consider the point of north-pole $n = (0, 0, 1)^T$ on the unit sphere. After the rotation, $R$, say $n$ is rotated to another point $p = (x, y, z)^T$; i.e., $p = Rn$. Let $\alpha$ denote the angle $\angle nO p$. Note that $\alpha$ determines the $z$-coordinate of $p$. To determine $x$-coordinate and $y$-coordinate of $p$, the point is rotated around the $z$-axis for the angle $\beta$ on the unit sphere. Note that there are infinitely many rotations that transform $n$ to $p$. The particular rotation $R$ can be decided by rotating all other points around the vector $p$ by the angle $\gamma$. It is not hard to verify that, in such a way, any rigid rotation transformation can be parameterized by the three-tuple $(\alpha, \beta, \gamma)$. Thus, we call a vector $p = (x, y, z)^T$ on the surface of the unit sphere a probe. Note that the movement of each probe is started from the northpole $(0, 0, 1)^T$ to other points in the sphere. The position of $p$ is decided by the parameters $(\alpha, \beta)$, and exact rotation is fixed by the self-rotation angle $\gamma$.

As a result, we reduce the problem of finding the good rotation matrix to the new problem of finding a good 3-parameter setting. The rotation matrix is thus characterized by just adjusting the 3 uniformly distributed parameters.

**Minimum bipartite matching**

We use the minimum bipartite matching algorithm to find the best matching between two sets of points to decide the best matching for the rmsd procedure. We adopted the Munkres [20,21] algorithm. In order to improve the efficiency of computation, we rewrite the Munkres algorithm, which is implemented on a public available perl module, by C language.

**VAST**

The structure alignment algorithm of VAST is based on aligning secondary structure elements using an algorithm from the field of graph theory. The output is a neighbors D list. It also contains the complete PDB representative structure comparison structure alignments and a structure superposition tool. The search space for alternative secondary structure elements depends on the length of proteins. All pairs of secondary structure elements (one from each structure) that have the same type are represented as nodes of a graph. This correspondence graph is then searched to find the maximal subgraph such that every node in the subgraph is connected to every other node in the subgraph and is not contained in any larger subgraph with this property. VAST has been used to compare all known PDB domains to each other. The results of this computation are included in NCBI’s Molecular Modeling Database at [http://www.ncbi.nlm.nih.gov/Structure/VAST/vast.html](http://www.ncbi.nlm.nih.gov/Structure/VAST/vast.html).

We use our protein structure alignment algorithm to refine the VAST result, Figure 1 and Figure 2 shows the example.

![Figure 1. The readjustment of the alignment suggested by the VAST system.](image1.png)

![Figure 2. The example of the initial alignment by VAST as the input and refined by our protein structure alignment algorithm. Alignment has been changed and improve the rmsd value.](image2.png)
Parametric adjustment with trigonometric series

In the trigonometric series estimation algorithm [22], the three parameters (rotational angles) are assumed to be independent. Three parameters are adjusted one by one, and the estimation function corresponding to the affected rmsd values is subsequently better estimated by increasing the curvature degree of the estimated function. The trigonometric series function is described as the following:

\[ f(\theta) = C_1 + C_2 \cos \pi \theta + C_3 \sin \pi \theta \\
+ C_4 \cos 2\pi \theta + C_5 \sin 2\pi \theta \\
+ C_6 \cos 3\pi \theta + C_7 \sin 3\pi \theta + ... \\
+ C_{2k} \cos k\pi \theta + C_{2k+1} \sin k\pi \theta. \]

Note that \( f(\theta) \) denotes the corresponding rmsd value with respect of one of the three parameters, \((\alpha, \beta, \gamma)\). The \( k \) usually reflects the numbers of local maximal points in the approximated curve.

The algorithm [22] improves the rmsd value between a protein structure pair by finding better alignment list. A set of experiments is designed to test the parameters; these adjusted parameters are set to perform the experiments over a thousand of protein pairs, which are uniformly random sampled from the PDB database. As the results shown that our methods improve the alignment computed by the VAST by an averaged improvement ratios about 11 \%. The results of Figure 3 demonstrate that the method of using 3D Euclidean distance minimum bipartite matching with trigonometric series estimated parametric searching scheme indeed improves existed known system like the VAST [23].

![Figure 3](image)

The visualization system shows each aligned pair by a stick; each stick represents the linkage between two C-alpha, representing the corresponding amino acids pair. The comparison result can be viewed by molecular biologists in our web service via the Cortona3D Viewer. As an example, Figure 4 shows the result of a sample PDB pair.

![Figure 4](image)

The computation needed to solve the structure comparison problem, like many scientific computation/simulation problem, is very time-consuming under cases of large structures and large number of paired structures. It is desirable to implement the system under massive parallel machines cluster, e.g., the grid-environment, to increase the throughput of the system.

2.2 Hadoop MapReduce framework

Hadoop is a software framework for coordinating computing nodes to process distributed data in parallel. Hadoop adopts the map/reduce parallel programming model, to develop parallel computing applications. The standard map/reduce mechanism has been applied in many successful Cloud computing service providers, such as Yahoo, Amazon EC2, IBM, Google and so on. An application developed by Map/Reduce is composed of Map stage and Reduce stage (optionally). Figure 5 illustrates the Map/Reduce framework. Input data will be split into smaller chunks corresponding to the number of Maps. Output of Map stage has the format of \(<key, value>\) pairs. Output from all Map nodes, \(<key, value>\) pairs, are classified by key before being distributed to Reduce stage. Reduce stage combines value by key. Output of Reduce stage are \(<key, value>\) pairs where each key is unique.

PDB2VRML

Our visualization tools take advantage of the PDB2VRML tool [24] to convert the molecular structure information produced by the protein alignment algorithm into the VRML format (.wrl).
Hadoop cluster includes a single master and multiple slave nodes. The master node consists of a jobtracker, tasktracker, namenode, and datanode. A slave node, as computing node, consists of a datanode and tasktracker. The jobtracker is the service within Hadoop that farms out Map/Reduce tasks to specific nodes in the cluster, ideally the nodes that have the data, or at least are in the same rack. A tasktracker is a node in the cluster that accepts tasks; Map, Reduce and Shuffle operations from a jobtracker.

Hadoop Distributed File System (HDFS) is the primary file system used by Hadoop framework. Each input file is split into data blocks that are distributed on datanodes. Hadoop also creates multiple replicas of data blocks and distributes them on datanodes throughout a cluster to enable reliable, extremely rapid computations. The namenode serves as both a directory namespace manager and a node metadata manager for the HDFS. There is a single namenode running in the HDFS architecture.

2.3 Protein structure analysis on Map/Reduce framework

Figure 5 illustrates the MapReduce framework for the protein structure analysis. Assume that the number of compute node is $N$ and the number of protein pairs lines is $P$ in the protein pairs list file, the $P$ lines will become $P$ map tasks and send to hadoop by streaming program. Each computing node receives a map and then performs the analysis work. When a node completes the map task, the node passes the score to the Reduce and receives next map task to compute unless the total map are finished. Generally, each node will be assigned to deal with $P/N$ maps.

Therefore, Reduce will have $P$ RMSD scores. The RMSD score is described in the previous section 2.A. The reduce operation writes the protein structure pair analysis scores line by line to the output file on HDFS.

3 The Web Service (Cloud) Platform

Our protein structure comparison web service can be seen as a SAAS (Software as a service) under general cloud computing terminology [25]. Here we describe its platform structure as the following.

Platform framework

Figure 5 shows our web service platform consisting of a Receiving site and a Processing site. The Receiving site serves as a regular web page service responsible for processing general user requests; while the Processing site is the Hadoop map/reduce platform that is ideal for massive parallel processing for huge amount of processing data.

There are three programs within the receiving site: First, `get-req.php` is a PHP program receiving user request; the second program, `collect.pl`, is a Perl program responsible for collecting PDB data from NCBI PDB before preparing the needed ready list. The third program, `dispatch.pl`, is responsible for sending the ready-list to the Processing site, and collecting processed results from the Processing site before sending out the final web page to the user. Note that `dispatch.pl` regular checks for ready-list and processed results from time to time.

There is one program within the processing site: `process.pl` is a Perl program responsible for receiving the ready-list from the Receiving site, and invoking protein comparison in hadoop platform. Note that the `process.pl` regularly checks for the ready-list from time to time.

4 The Experimental (Cloud) Platform

4.1 Experimental environment and data source

The experiments are performed on one NFS server and four IBM blade servers in the Providence University Cloud Computation Laboratory. Each server is equipped with two Quad-Core Intel Xeon 2.26GHz CPU, 24G RAM, and 296G disk under the Ubuntu version 10.4 with the virtualization platform KVM/QEMU. Under the current system environment, we create 8 virtual machines by KVM; each virtual machine is set to one core CPU, 2G RAM, and 30G disk running under the O.S. Ubuntu version 10.4 with Hadoop version 0.2 MapReduce platform. Each virtual machine is responsible for one map operation and one reduce operation. The total number of the map/reduce operations is up to 8 respectively.

The Protein structure data sources are gathered from the World Wide Protein Data Bank (http://www.wwpdb.org), which is a PDB format content with 4-letter entry name. The Protein Data Bank maintains this single archive of macromolecular structural data freely and publicly available to the global community. These protein structure data are downloaded from the wwPDB's ftp server.
(ftp://ftp.wwpdb.org/), where the number of protein structure data is 80,402. These data are treated as the testing data for our experiments.

4.2 Web System

Our system is accessible through the web system at sites.google.com/site/pucspssae/, bioinfo.cs.pu.edu.tw/bioinfo/3Dvisual.html and bioinfo.cs.pu.edu.tw/bioinfo/3Dvisualvast.html for molecular biologists interested in three dimensional protein structures alignment/comparison problems. The user of our web service usually provides two protein PDB IDs. Our system fetches the protein data from the wwPDB data base, and performs the desired protein structure comparison algorithms, chosen by the user.

To avoid the time delay, our web service provides user access keys for user to check the result later on after they submit the desired query. User can come back and check the result once tasks are finished by the system.

Figure 6. The Protein structure analysis by the hadoop platform with Map/reduce frameworks.

Different initial alignment methods can be chosen as options to refine better alignment results. Currently there are three types of initial alignments: cut1 [23], VAST, or the user’s own specification. It means that users can upload their own initial alignments to use the web service to obtain better refined alignments. Many other types of initial methods [23] will be provided in the web service in the near future.

Figure 7. Examples of initial options in the web service.

Performance

The diagram of Figure 8 shows the web site performance function; the x-axis depicts log2(n), average lengths of PDB pairs, and the y-axis depicts log2(T), the corresponding computation times. There are 10 points, and each point represents on the rectangular coordinate is a pair (n, T(n)).

The linear regression line, shown in red, represents the trend of the growing function; the regression analysis suggests that T(n) ≈ 0.0173 · n1.0216 msec. The system time complexity is approximately O(n^3), which is consistent with the previous known theoretic time complexity of minimum bipartite matching algorithm by Munkres [20,21] as the main system bottleneck.

![Figure 8. Time complexity of the web service T(n) versus different length of protein pairs.](image)

5 Concluding Remarks

In this paper, we have proposed visualization tools for pairwise protein structures alignment and compared the performances between the sequential version and the parallelized map/reduce version of our previously proposed protein structure alignment algorithms. 3D protein structures alignment is useful in analyzing and classifying these protein structures in order to discover the structural relationships with protein functions. Our methods provide the possibility to establish relevant relationships, even between protein families distinct in terms of sequence comparison alone.

In the future, we will further investigate algorithmic methods for discovering bioinformatics functions, and try to parallelize these methods to provide more perspectives for biologists to improve the performance of these computation frameworks for analyzing the increasingly huge bioinformatics data.
6 Acknowledgment

This research was partially supported by the National Science Council under the Grants NSC-99-2632-E-126-001-MY3

References


[24] Horst Vollhardt. This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself. 1998.