CORE: COmmon REgion extension-based protein structure alignment for producing multiple solution

Woo-Cheol Kim, Sanghyun Park and Jung-Im Won

College of Information Sciences and Technology, The Pennsylvania State University
IST Building, University Park, PA, 16802, USA
wxk11@psu.edu

Department of Computer Science, Yonsei University
50 Yonsei-ro, Seodaemun-gu, Seoul 120-749, Korea
sanghyun@cs.yonsei.ac.kr

Research Center of Information and Electronic Engineering, Hallym University
1 Hallymdaehak-gil, Chuncheon, Gangwon-do 200-702, Korea
jiwon@hallym. ac.kr

Over the past several decades, biologists have conducted numerous studies examining both general and specific functions of proteins. Generally, if similarities in either the structure or sequence of amino acids exist for two proteins, then a common biological function is expected. Protein function is determined primarily based on the structure rather than the sequence of amino acids. The algorithm for Protein Structure Alignment is an essential tool for the research. The quality of the algorithm depends on a quality of the similarity measure that is used. This is because the similarity measure is an objective function used to determine the best alignment. However, none of similarity measures became golden standard because their individual strength and weakness. It was derived excessive filtering to find a single solution. In this paper, we introduce a new strategy that finds not a single solution, but multiple solutions with different lengths. This method has obvious benefits of high quality alignment. However, this novel method leads to a new problem that the running time for this method is considerably longer than that for methods that find only a single solution. To address this problem, we propose algorithms that can locate a COmmon REgion (CORE) of multiple solutions candidates, and can then extend the CORE into multiple solutions. Because the CORE can be defined from a final alignment, we introduce CORE* that is similar to CORE and propose an algorithm to identify the CORE*. By adopting CORE* and DP, our proposed method produces multiple solutions of various lengths with higher accuracy than previous methods.

Keywords: protein structure, structure alignment, sequence alignment, similarity search

1. INTRODUCTION

Over the past several decades, biologists have conducted numerous studies examining both general and specific functions of proteins [1]. However, due to the time and effort required for the experimental methods employed by biologists, there are numerous limitations to these approaches exist in analyzing protein functions. As such, automated computer systems have recently been developed to analyze the functions of multiple proteins simultaneously [2].

Generally, if similarities in either the structure or sequence of amino acids exist for two proteins, then a common biological function is expected [3]. Protein function is determined primarily based on the structure rather than the sequence of the amino acids that compose it [4]. The fact that the sequence of amino acids can be mutated into other forms during evolutionary processes while the structure generally remains intact is evidence of this [5]. In this context [6], it has been shown that replicate proteins can be artificially constructed from existing proteins when the two proteins have a similar function and structure but a dissimilar sequence of amino acids. That is, the structure of functionally related proteins provides additional insight into their functional mechanisms and this knowledge has been successfully applied to the functional annotation of proteins whose structure has been previously identified [7,8]

The Structure Alignment Problem (SAP) is the term applied to the need to find the *superposition* and the *transformation* that will allow the backbones of two proteins to be aligned resulting in the highest similarity score. The superposition is a mapping that is completed between the residues of two proteins, and the transformation is the combination of the rotation and translation of the proteins. The SAP is a non-deterministic polynomial-time hard (NP-hard) problem [9], and thus, most methods used to address this problem employ a heuristic approach [10,11].

A general framework for the development of an algorithm for the SAP consists of the following four steps [12]: 1) represent the structure of the two proteins to be aligned in spatial coordinates; 2) align the two protein structures; 3) optimize the alignment; and 4) evaluate the statistical significance of the alignment using Z-scores or other methods.

The quality of the algorithm depends on a quality of the similarity measure that is used. This is because the similarity measure is an objective function used to determine the best alignment of the two proteins [13]. Most SAP algorithms employ the cRMSD (coordinated Root Mean Square Deviation)-based similarity measure. When using the cRMSD as similarity measure solely, it is difficult to intuitively judge the level of similarity of the two protein structures [14]. Better alignment is expected when the alignment length is long and the cRMSD is small. However, the alignment length and the cRMSD are positively correlated, such that when the alignment length became longer, the cRMSD gets larger. As such, many studies on SAP attempt to balance the two parameters through the use of heuristics or statistical methods. Although several usable SAP algorithms have been developed, none have resulted in a golden or optimal rate. In particular, it is difficult to determine which would be the more accurate alignment in the following example: (85, 2.8Å) or (102, 3.2Å), where (alignment length, cRMSD).

In this paper, a new strategy is employed that finds not a single solution, but multiple solutions with different lengths. This method has obvious benefits in that it does not require that the two parameters be balanced. However, this novel method leads to a new problem. Specifically, the running time for this method is considerably longer than that for methods that find only a single solution. To address this problem, we propose algorithms that can locate a common region of multiple solutions.

2. RELATED WORKS

Protein alignment algorithms can be classified into three categories according to the type of data that is used in the algorithms [15]. Specifically, the algorithms in the first, second and third categories use the C_α atom, Secondary Structure Element (SSE), and geometric hashing, respectively.

For protein alignment algorithms in the first category, the simplest method involves using the C_α atom in dynamic programming. The best-known of these methods are DALI [5] and CE [16]. Double Dynamic Programming [17], Iterative Dynamic Programming [18] and MINRMS [19] are based on multiple forms of dynamic programming.

The DALI aligns protein structures through the use of distance matrices. A distance matrix is an *n*n* matrix, representing the distance between each pair of residues of the underlying protein structure. Specifically, the cell at the *i*-th row and the *j*-th column of the distance matrix denotes the distance between the *i*-th residue and the *j*-th residue in the protein structure. Proteins with similar structures produce similar distance matrices. To align the structures of two proteins, DALI compares their distance matrices. To reduce the processing time for this type of alignment, DALI divides the distance matrix of each protein structure into segments and identifies segment pairs with similar distance patterns. Then, the overlapping segment pairs identified in the previous step are combined into larger segment pairs to maximize the similarity score.

Rather than using distance matrices, the CE approach divides protein structures into fragments and finds pairs of these fragments, called an Aligned Fragment Pair (AFP), which display high levels of similarity. Next, similar to the segment pairs found using the DALI, the AFPs are extended into a final alignment.

The computational cost of methods that are based on dynamic programming, such as DALI and CE, depends upon the length of the protein structures to be aligned. Both DALI and CE have adopted a heuristic approach to reduce computational cost by treating a certain number of C_α atoms as a single processing unit. However, these methods do so without considering the similarity of the protein structures to be aligned. Therefore, even if the two protein structures being aligned have a high level of similarity, each protein structure has to be broken into multiple fragments that are subsequently compared with the fragments of the other protein. One limitation of these methods is that they require a significant amount of time even when the protein structures are analogous.

Lotan and Schwarzer proposed an algorithm [20] to minimize the alignment time by reducing the number of C_α atoms prior to the alignment process. This algorithm uses a small number of C_α atoms during the alignment process, and thus, reduces the alignment time. This method can determine the similarity of aligned protein structures. However, this method is unable to determine which of the residues are responsible for the alignment.

One method that is representative of the second category of protein alignment algorithms, called VAST, uses SSE [21]. VAST is a hierarchical alignment algorithm. The algorithm initially aligns the two proteins using only their SSE, which is the higher structure data than the C_α atoms. Next, the aligned result is used as a superposition that is extended into the C_α atom level. One of the relative strengths of these algorithms is that it allows for the superposition to be more quickly identified through the use of SSEs rather than C_α atoms, thus reducing the total alignment time. However, in this process, there are still proteins whose C_α atoms are identified although the SSEs are not, and this limits the usage of this algorithm. Information about the C_α atoms can be obtained through Nuclear Magnetic Resonance (NMR) spectroscopy and/or X-ray crystallography. Typical programs used to locate SSEs include DSSP [22] and STRIDE [23].

The third category of algorithms uses geometric hashing, which converts reference frames into hash values and stores them as a hash table. Protein structures are then aligned using these hash values as frames of reference. 3-D lookup [24] is another representative algorithm used in the alignment of protein structures, and this method employs the SSE as a reference frame. Additionally, Nussinov and Wolfson [25] have proposed a protein structure alignment algorithm that employs the C_α atom as a reference frame [26]. However, the accuracy of these methods is low as only the predefined reference frames are used during the alignment process.

Recently, Erdmann [27] proposed a protein structure alignment algorithm. To align the protein structure, this algorithm finds the helix and intersections of each protein structure by employing knot theory and geometric convolution. Although two protein structures may not be similar as a whole, this algorithm determines that they are analogous if the proteins have a similar helix or intersection.

3. PROBLEM DEFINITION

In this section, the SAP is defined as a numerical formula. The protein structure, *S*, is given with the complete set of *x*, *y*, *z* coordinates for all atoms. This representation can be further reduced to the α-carbon backbone atoms, $S = \langle r_i \rangle_{i=1}^{n}$, where *n* is the number of amino acids. The amino acids are ordered according to the preexisting order on the protein chain. The substructure of protein, $u = \langle q_i \rangle_{i=1}^{k}, u \subset S$, is the structure that is a subset of atoms in *S*. u_k and P_k denote the substructure and the set of substructures with the same length, where *k* is the length of the substructure.

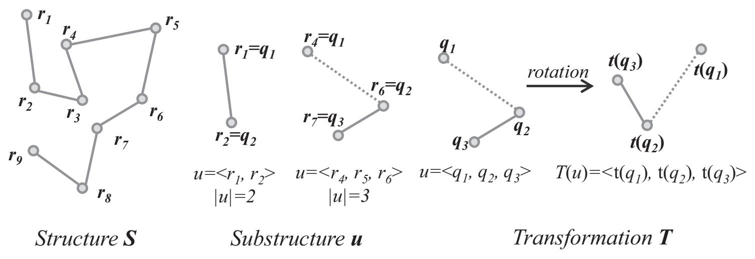


Figure 1. Protein structure, S, sub-structure, u, and transformation, T.

In this paper, structure alignment refers not to the global alignment but to the local alignment. Therefore, a similarity measure for substructures is needed. It is denoted by $M(u^a,u^b|F)$, where F is a similarity function that expresses the similarity between u^a and u^b. Here a larger F implies that the pair is more similar. Although the cRMSD and dRMSD (distance Root Mean Square Deviation) are distance functions, they are adequate representations of F because they have been converted to a similarity function through a conversion formula, $\text{sim}=1/(1+\text{dist})$. Specifically, our proposed method uses cRMSD as the F, and M is represented as follows:

$$\text{M}(u^a,u^b|F)=1/\left(1+\text{cRMSD}(u^a,u^b)\right)=1/\left(1+\sqrt{\frac{\sum_{i=1}^{k}d_i}{k}}\right), k=|u^a|=|u^b|, d_i=|q_i^a-q_i^b|$$

Based on this definition, the SAP is defined as follows:

$$\text{SAP}(S^a,S^b)=(u^{*a},u^{*b})=\underset{u^a,u^b}{\text{argmax}}\,\text{M}(u^a,u^b\mid F), u^a\subset S^a, u^b\subset S^b, |u^a|=|u^b|$$

The formulas outlined above are explained through the following steps. First, all substructure pairs (u^a, u^b) on S^a and S^b are populated. Second, M for every pair is computed, and the pair (u^{*a},u^{*b}) for which M is the highest is chosen as the final result.

However, this formula needs to be modified since it is possible for S to be rotated (*rotation*) or translated (*translation*) in 3-dimensional space, and these actions are called *transformation*. Therefore, even when the same pair is aligned, its M may have a very different value as a result of transformation. For this reason, the SAP is redefined with the following formula including transformation, T.

$$\text{SAP}(S^a,S^b)=(u^{*a},u^{*b},T^*)=\underset{u^a,u^b,T}{\arg\max}\,\text{M}(T(u^a),u^b\mid F),\quad u^a\subset S^a, u^b\subset S^b, |u^a|=|u^b|$$

, where (u^{*a},u^{*b}) and T^* indicate a superposition and a transformation, respectively.

4. SIMILARITY MEASURE

The cRMSD has a critical weakness as a similarity function as it does not reflect the alignment length. For example, when there are two alignments, $|u'^a| = |u'^b| = 21$ with cRMSD = 0.2Å and $|u''^a| = |u''^b| = 48$ with cRMSD = 0.3Å, the former is the better alignment according to the cRMSD. However, in this example, the latter is clearly the better alignment if the alignment length is considered in addition to the cRMSD.

Due to this problem, the SAP algorithms using cRMSD have proposed similarity measures that combine cRMSD and alignment length. To find an optimum or golden rule for the combination of cRMSD and alignment length, many researchers have applied heuristics that have been identified based on the experience of these researchers and therefore reflect the subjective views of the particular researcher. Thus, a particular user is unlikely to identify the best alignment by using another researcher's algorithm if their perspectives differ.

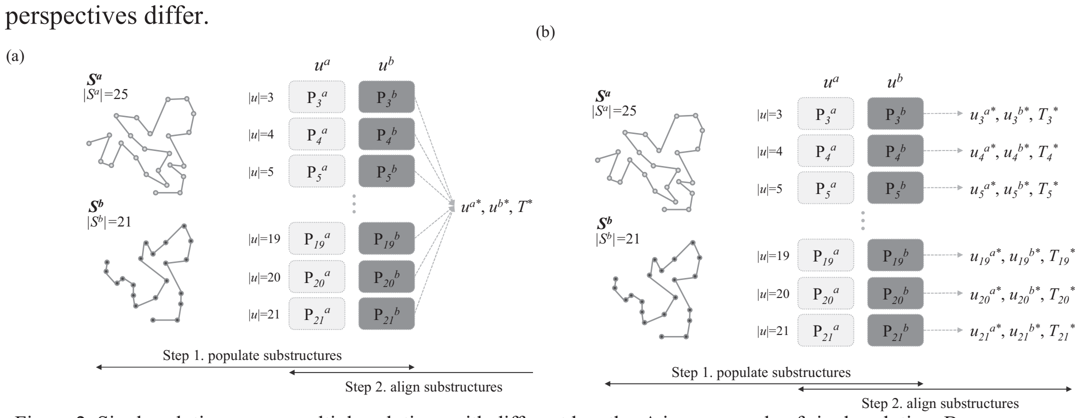

(a)
S^a
$|S^a|=25$
S^b
$|S^b|=21$
u^a
u^b
$|u|=3$
$|u|=4$
$|u|=5$
$|u|=19$
$|u|=20$
$|u|=21$
P_3^a P_3^b
P_4^a P_4^b
P_5^a P_5^b
P_{19}^a P_{19}^b
P_{20}^a P_{20}^b
P_{21}^a P_{21}^b
u^{a*}, u^{b*}, T^*
Step 1. populate substructures
Step 2. align substructures
(b)
$u_3^{a*}, u_3^{b*}, T_3^*$
$u_4^{a*}, u_4^{b*}, T_4^*$
$u_5^{a*}, u_5^{b*}, T_5^*$
$u_{19}^{a*}, u_{19}^{b*}, T_{19}^*$
$u_{20}^{a*}, u_{20}^{b*}, T_{20}^*$
$u_{21}^{a*}, u_{21}^{b*}, T_{21}^*$

Figure 2. Single solution versus multiple solutions with different lengths: A is an example of single solution. B is an example of multiple solutions with various lengths

In order to address the problem of the cRMSD-based SAP, a straightforward solution is proposed that finds not a single solution but multiple solutions, each of which is the best alignment for each individual alignment length.

$$\mathrm{SAP}\left(S^a, S^b\right) = \bigcup_{k=1}^{N} \mathrm{SAP}\left(P_k^a, P_k^b\right), \quad \mathrm{N} = \min\left(\left|S^a\right|, \left|S^b\right|\right)$$

$$= \left(u_1^{a*}, u_1^{b*}, T_1^*\right) \cup \left(u_2^{a*}, u_2^{b*}, T_2^*\right) \cup \cdots \cup \left(u_i^{a*}, u_i^{b*}, T_i^*\right) \cup \cdots \cup \left(u_N^{a*}, u_N^{b*}, T_N^*\right)$$

The method for finding multiple solutions is a simple and objective way to address the aforementioned problem. However, the amount of time required for this technique will be N-times greater than that of the methods that find a single solution if a naïve algorithm is used to find the multiple solutions (where N is the number of alignments). Therefore, in order to reduce the time-complexity, the CORE-based alignment is introduced in the next section.

5. CORE AND CORE*

To reduce the processing time required to find multiple solutions, the ideas proposed in [28] were utilized. Heuristic-based algorithms typically find a local optimum not a global optimum. A common region with long length residues exists between the local optima [28]. Thus, if an algorithm is developed to locate this common region, the time-complexity for finding multiple solutions may be reduced. Therefore, the common region is defined as the CORE, and an algorithm to find the CORE and to extend the CORE into multiple solutions is proposed herein.

(a) $u^a = < r^a_1, r^a_3, r^a_4, r^a_5, r^a_6, r^a_9 >$
$u^b = < r^b_3, r^b_4, r^b_5, r^b_6, r^b_7, r^b_8 >$

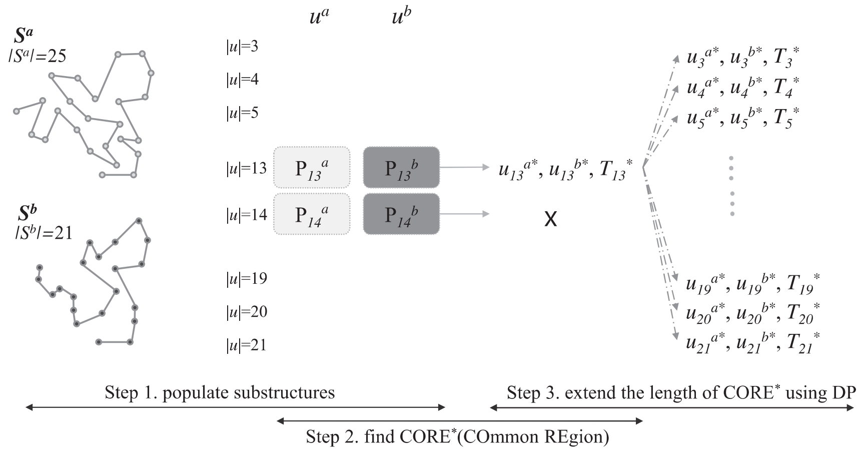
(b) $u^a = < r^a_2, r^a_3, r^a_4, r^a_5, r^a_6, r^a_9, r^a_{10} >$
$u^b = < r^b_2, r^b_4, r^b_5, r^b_6, r^b_7, r^b_8, r^b_9 >$

(c) $u^a = < r^a_2, r^a_3, r^a_4, r^a_5, r^a_6, r^a_9 >$
$u^b = < r^b_3, r^b_4, r^b_5, r^b_6, r^b_7, r^b_8 >$

(d) $u^a = < r^a_3, r^a_4, r^a_5, r^a_6, r^a_{11}, r^a_{12} >$
$u^b = < r^b_4, r^b_5, r^b_6, r^b_7, r^b_{11}, r^b_{12} >$

(e)
$u^a = < r^a_3, r^a_4, r^a_5, r^a_6 >$
$u^b = < r^b_4, r^b_5, r^b_6, r^b_7 >$

Figure 3. Various local optima produced by various SAP algorithms: Each alignment, A, B, C and D, has characteristics that reflect the heuristics used in SAP algorithms. E is the common region of A, B, C and D.



u^a
u^b
S^a
$|S^a|=25$
S^b
$|S^b|=21$
$|u|=3$
$|u|=4$
$|u|=5$
$|u|=13$
$|u|=14$
$|u|=19$
$|u|=20$
$|u|=21$
P_{13}^a
P_{13}^b
P_{14}^a
P_{14}^b
$u_{13}^{a*}, u_{13}^{b*}, T_{13}^*$
X
$u_3^{a*}, u_3^{b*}, T_3^*$
$u_4^{a*}, u_4^{b*}, T_4^*$
$u_5^{a*}, u_5^{b*}, T_5^*$
$u_{19}^{a*}, u_{19}^{b*}, T_{19}^*$
$u_{20}^{a*}, u_{20}^{b*}, T_{20}^*$
$u_{21}^{a*}, u_{21}^{b*}, T_{21}^*$
Step 1. populate substructures
Step 3. extend the length of CORE* using DP
Step 2. find CORE*(COmmon REgion)

Figure 4. Proposed algorithm.

Finding the CORE (step 2 in Figure 4) is the most important step in the proposed algorithm. According to the analysis of alignments produced by various SAP algorithms, two characteristics of the final alignment were identified. One of these characteristics is that the number of fragment-pairs in the alignment is small. The fragment is defined as the *f* that is a substructure and that satisfies the order constraint: $q_m=r_o$, $q_{m+1}=r_{o+1}$, and the fragment-pair that is a pair of *f*. For example, the number of fragment-pairs in the alignment for which the length is around 100 is approximately 12.

(a) $S^a = < r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8 >$ (b) $u^a = < r_2, r_3, r_4, r_7, r_8 >$ (c) $u^a = < r_2, r_3, r_4, r_7, r_8 >$
$S^b = < r_1, r_2, r_3, r_4, r_5, r_6, r_7 >$ $u^b = < r_1, r_2, r_5, r_6, r_7 >$ $u^b = < r_1, r_2, r_5, r_6, r_7 >$

Figure 5. An example of the fragmentation of a final alignment: A. Protein structures, S^a and S^b, consist of 8 and

7 amino acids (or residues), respectively. B is a structure alignment for S^a and S^b. In B, although both u^a and u^b have two fragments, the fragmented position is not the same. As a result, the number of fragment-pairs in the alignment is 3.

The other characteristic is that the lengths of the fragment-pairs in the alignment do not follow a normal distribution. The lengths of most fragment-pairs are much smaller than the average length of the fragment-pairs, and the longest fragment-pair forms a large portion of the alignment. For example, when the average fragment-pair length is 8, it is common that the fragment-pair length pattern in the final alignment is 14-26-2-2-6 rather than 10-8-6-7-9. When comparing the alignments of highly homologous proteins (e.g., the *family* level of the hierarchical SCOP structural classification database [29]) the average length of an alignment and the average lengths of the longest fragment-pair were 127 and 47, respectively.

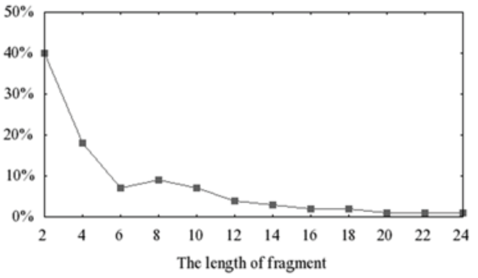


Figure 6. A graph showing the distribution of the fragment-pair lengths: The length of most fragment-pairs is smaller than the average (=8).

Definition 1. CORE

Given an alignment (u^{a*}, u^{b*}) *for two protein structures,* S^a *and* S^b*, let* $(u^{'a*}, u^{'b*})$ *be the aligned fragment-pair that is satisfied* $(u^{'a*}, u^{'b*})$, $u^{'a*} \subset u^{a*}$ *and* $u^{'b*} \subset u^{b*}$ *and* $u^{'a*}$ *and* $u^{'b*}$ *are fragments. CORE denotes the longest aligned fragment-pair.*

According to Definition 1, the CORE can be found only from a final alignment. However, the CORE must be used as seed to find an alignment. As a solution to this problem, we introduce CORE* that is similar to CORE and propose an algorithm to identify the CORE*.

Definition 2. CORE*

Given two fragments, f^a *and* f^b*, from structures* S^a *and* S^b*, respectively, let* f^a *and* f^b *be the aligned fragment-pair candidates that satisfy the condition that the similarity score of their alignment is larger than the user-defined threshold, T. CORE* denotes the longest aligned fragment-pair candidate.*

As discussed above, the most significant characteristic of CORE is that its length is significantly longer than the average length of the fragment-pairs. That is, if the longest fragment-pair is identified from those with a similarity level greater than that of the average pair, it is probably the same as that of CORE. This observation is more accurate when the length of the CORE is longer. In preliminary experiment with a real dataset, 97% of CORE*s of which length is over 32 were identical to those identified with CORE. The remaining 3% of the CORE* findings only have 2 mismatches.

A naïve procedure to find CORE* is composed of the following 4 steps: 1) generate all fragment-pairs from S^a and S^b; 2) compute M for all fragment-pairs; 3) remove fragment-pairs for which the M is smaller than the user threshold; and 4) choose the longest fragment-pair among the survivors of the previous steps as the CORE*. However, this procedure is time-consuming as the number of generated fragment-pairs is quite large. For example, the number of fragment-pairs from $S^a(|S^a|=120)$ and $S^b(|S^b|=100)$ is 439,350. Therefore, the following two major heuristics are adopted to increase the speed of the verification process for the fragment-pairs.

One heuristic is that the verification process proceeds from the longer fragment-pair to the shorter fragment-pair as it is desirable to identify the longest survivor. In particular, the practical maximum length of CORE*, not the theoretical maximum length; the smaller length of S^a and S^b, is used as the beginning length for this process. As an example, when aligning two structures, S^a and S^b, the lengths of which are 120 and 100, respectively, the practical maximum length of CORE is no longer than 43. Nonetheless, the theoretical maximum length of CORE* is 100. This implies that it is needless to compute the M of fragment pairs over a length of 43.

The other heuristic involves changing the similarity function from cRMSD to dRMSD and L_∞, in order to increase the speed of the computation of M. CORE* is found not by the exact value of M, but by knowing whether the value of M is larger than the user-defined threshold. According to the analysis, it was determined that there was a high positive correlation between cRMSD, dRMSD, and L_∞ when the length of CORE* is long enough. The computation time for cRMSD is considerably longer than that for dRMSD and L_∞ because cRMSD requires *transformation*. This appears to be true when L_∞ is used as F, and it is faster to check whether L_∞ of fragment-pair is greater than the user-defined threshold.

$$\text{dRMSD}\left(u^a,u^b\right)=\sqrt{\frac{\sum_{i=1}^{|A|}\sum_{j=1}^{|A|}\left(d_{ij}^a-d_{ij}^b\right)^2}{|A|}},d_{ij}=\left|r_i-r_j\right|$$

$$L_\infty\left(u^a,u^b\right)=\max_d\sum_{i=1}^{k}d_i,\quad k=\left|u^a\right|=\left|u^b\right|,d_i=\left|q_i^a-q_i^b\right|$$

The above-obtained CORE* is a seed for the final alignment. However, the CORE* is too short when compared to the final alignment. Our analysis determined that its coverage is not higher than 40% of the final alignment. The DP is known to have the advantage of accuracy but the disadvantage of time-complexity. We applied the DP to get longer CORE*.

7. EXPERIMENTS

Experiments were conducted using a variety of protein structure data in Protein Data Bank(PDB) [30] to demonstrate the superiority of the proposed algorithm. First, the way that the number of CORE*s affects performance was examined by comparing the alignments according to the number of CORE*s. Second, performance when the refinement process was conducted compared to when it was not was examined to verify the necessity of the refinement process.

Two parameters were used in the experiments. The first was that the length of the fragment pair was used as the start length of verification progress when finding CORE*. It was revealed that more than 99% of the alignments had COREs with a length of less than 38. Therefore, the alignment was conducted on the fragment pairs that had a length of 38. The second parameter used was the similarity score. The thresholds for dRMSD and L_∞ were used to decide whether or not the fragment-pairs could be the CORE*. After analysing the CORE of alignments, the thresholds were decided to 9.41Å and 4.05Å, respectively.

7.1 The processing time and accuracy according to the number of the appended COREs*

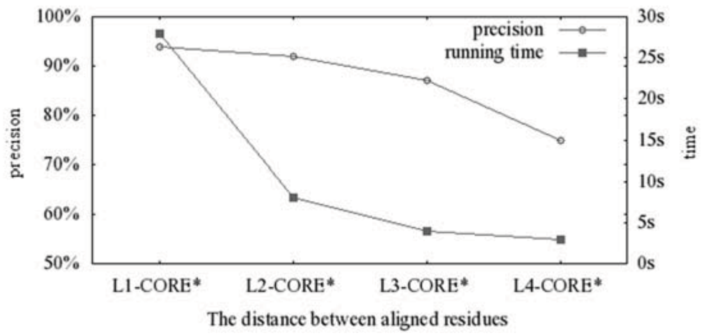


Figure 7. The processing time and accuracy according to the number of appended CORE*s used for seeds.

In the process of finding seeds, the more CORE*s that were used, the longer the average length of the seeds was. Consequently, this affected the accuracy of the alignment and processing time. As the accuracy decreased, the processing time was reduced according to the increase in the number of COREs used, as depicted in Figure 7.

7.2 The accuracy after refining the COREs.

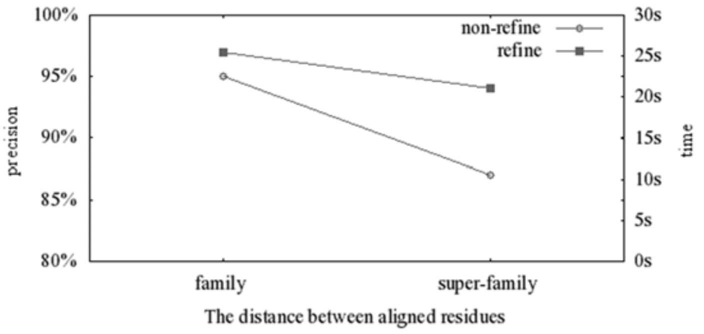


Figure 8. The accuracy according to the refinement process of the family and super-family.

After locating seeds, the refinement process was executed. This eliminated the unnecessary mismatches of the seeds. This was particularly true as shown in the example in Figure 8.

8. CONCLUSION

The most important aspect of designing algorithms for the protein structures alignment is the balancing of the two parameters, the alignment length and the similarity score (i.e., the cRMSD). In this case, a better result is expected, when the alignment length is long and cRMSD is small. However, research has shown a positive correlation between the alignment length and the cRMSD such that increasing the alignment length is associated with an increase in the cRMSD. Therefore, the algorithm must incorporate the proper alignments that best balance the two parameters. However, this choice depends on what alignments are identified in each case.

The contributions of this paper are summarized as follows: 1) we proposed a method to produce multiple solutions of protein structures of different lengths and reduce the processing time by adopting CORE*. 2) Our algorithm makes it possible for users to visually choose the alignment they want from multiple solutions using a graph-based representation.

ACKNOWLEDGEMENTS

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0004815).

REFERENCES

[1] Ginalski K, Grishin NV, Godzik A, Rychlewski L (2005) Practical lessons from protein structure prediction. Nucleic Acids Research 33: 1874-1891.

[2] Roytberg M, Gambin A, Noe L, Lasota S, Furletova E, et al. (2009) On subset seeds for protein alignment. IEEE/ACM Trans Comput Biol Bioinform 6: 483-494.

[3] Mayr G, Domingues FS, Lackner P (2007) Comparative analysis of protein structure alignments. BMC Struct Biol 7: 50.

[4] Zhang Y (2009) Protein structure prediction: when is it useful? Current Opinion in Structural Biology 19: 145-155.

[5] Holm L, Sander C (1993) Protein structure comparison by alignment of distance matrices. J Mol Biol 233: 123-138.

[6] Dahiyat BI, Mayo SL (1997) De novo protein design: fully automated sequence selection. Science 278: 82-87.

[7] Yakunin AF, Yee AA, Savchenko A, Edwards AM, Arrowsmith CH (2004) Structural proteomics: a tool for genome annotation. Curr Opin Chem Biol 8: 42-48.

[8] Menke M, Berger B, Cowen L (2008) Matt: local flexibility aids protein multiple structure alignment. PLoS Comput Biol 4: e10.

[9] Sippl MJ, Wiederstein M (2008) A note on difficult structure alignment problems. Bioinformatics 24: 426-427.

[10] Chen L, Zhou T, Tang Y (2005) Protein structure alignment by deterministic annealing. Bioinformatics 21: 51-62.

[11] Glasgow J, Kuo T, Davies J (2006) Protein structure from contact maps: A case-based reasoning approach. Information Systems Frontiers 8: 29-36.

[12] Bourne PE, Weissig H (2003) Structural Bioinformatics. New York: John Wiley.

[13] Bhattacharya S, Bhattacharyya C, Chandra NR (2007) Comparison of protein structures by growing neighborhood alignments. BMC Bioinformatics 8: 77.

[14] Kolbeck B, May P, Schmidt-Goenner T, Steinke T, Knapp EW (2006) Connectivity independent protein-structure alignment: a hierarchical approach. BMC Bioinformatics 7: 510.

[15] Eidhammer I, Jonassen I (2001) Protein structure comparison and structure patterns – an algorithmic approach. Proc Int Conf Intell Syst Mol Biol.

[16] Shindyalov IN, Bourne PE (1998) Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Engineering 11: 739-747.

[17] Taylor WR, Orengo CA (1989) Protein structure alignment. J Mol Biol 208: 1-22.

[18] Taylor WR (1999) Protein structure comparison using iterated double dynamic programming. Protein Science 8: 654-665.

[19] Jewett AI, Huang CC, Ferrin TE (2003) MINRMS: an efficient algorithm for determining protein structure similarity using root-mean-squared-distance. Bioinformatics 19: 625-634.

[20] Lotan I, Schwarzer F (2004) Approximation of protein structure for fast similarity measures. Journal of Computational Biology 11: 299-317.

[21] Gibrat JF, Madej T, Bryant SH (1996) Surprising similarities in structure comparison. Current Opinion in Structural Biology 6: 377-385.

[22] Kabsch W, Sander C (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22: 2577-2637.

[23] Frishman D, Argos P (1995) Knowledge-based protein secondary structure assignment. Proteins-Structure Function and Genetics 23: 566-579.

[24] Holm L, Sander C (1995) 3-D lookup: fast protein structure database searches at 90% reliability. Proc Int Conf Intell Syst Mol Biol 3: 179-187.

[25] Nussinov R, Wolfson HJ (1991) Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques. Proc Natl Acad Sci U S A 88: 10495-10499.