New Hashing Techniques for Three-Dimensional Protein Structures

Tatsuya Akutsu[1]
akutsu@cs.gunma-u.ac.jp

Kentaro Onizuka[2]
onizuka@mrit.mei.co.jp

Masato Ishikawa[3]
ishikawa@icot.or.jp

[1] Department of Computer Science, Gunma University
1-5-1 Tenjin, Kiryu, Gunma 376 Japan

[2] Human Interface Research Laboratory, Matsusita Research Institute Tokyo, Inc., 3-10-1 Higashimita, Tama-ku, Kawasaki 214 Japan

[3] Institute for New Generation Computer Technology
1-4-28 Mita, Minato-ku, Tokyo 108 Japan

Abstract

This paper describes new methods to evaluate the structural similarity of proteins. In each method, a hash vector is associated with each fixed-length fragment of protein structure, where the following desirable property is theoretically proved: if the root mean square deviation between two fragments is small, then the distance between the hash vectors is small. Using the hash vectors, searching for similar protein structures can be done quickly. The methods were compared with the previous methods using PDB data, and were shown to be much faster.

1 Introduction

Comparing three-dimensional (3D) protein structures is important for genome informatics. Indeed, a lot of methods have been proposed to compare 3D protein structures [2, 8, 9, 11, 12, 15, 16, 17]. However, few methods can be used for quick searching for similar structures among a large number of protein structures. Quick searching for similar structures is very important for interactive uses of protein structure database systems since the number of proteins, for which 3D structures are known, is several hundreds now and it grows year by year.

[1]阿久津達也：群馬大学工学部情報工学科，〒 376 群馬県桐生市天神町 1-5-1
[2]鬼塚健太郎：松下技研 ヒューマンインタフェース研究所, 〒 214 川崎市多摩区東三田 3-10-1
[3]石川幹人：新世代コンピュータ技術開発機構, 〒 214 東京都港区三田 1-4-28

Here, we briefly review previous work. While several problems can be considered for searching for similar structures, this paper focuses on the following problem (the *substructure search problem*): given a fragment structure P and a positive real number δ, find all proteins in a database each of which contains at least a fragment R such that the *root mean square deviation* (*rmsd*, in short) between P and R does not exceed δ. Nussinov and Wolfson's method [8] based on the geometric hashing technique may be used for quick substructure searching. However, it requires long preprocessing time and large memory space. Other types of geometric hashing techniques have also been proposed in computer vision [6, 7], while none of them seems to be suited to this problem. An FFT(Fast Fourier Transform)-based algorithm, which was developed for computer vision by Schwartz and Sharir [13], may be used. Although their algorithm is elegant and efficient, it is not practical for typical sizes of fragment structures. Indeed, experimental results show that it is better than a naive method only when the number of residues is more than $200 \sim 300$ [14].

In the previous paper, one of the authors proposed a method for quick substructure searching, named the *least-squares hashing method* [1]. It is quite different from the geometric hashing technique used by Nussinov and Wolfson [8]. In the least-squares hashing method, a vector of real numbers is associated with each fixed length fragment of protein structure. It works very well in most cases. But, it sometimes fails to find similar substructures because the following property does not necessarily hold: if *rmsd* between two fragments is small, the distance between the associated hash vectors is small. Thus, we have developed new hashing methods, in which the above property is theoretically proved. As far as we know, the proposed hash vectors are the first ones for which the above property is theoretically proved. Moreover, experimental results using PDB (Protein Data Bank) data [3] show that the new methods are much faster than the naive method and the least-squares hashing method.

2 Preliminaries

In this paper, protein structures are treated as follows. As we are only interested in representing an outline of 3D structure, we follow the common procedure of ignoring side chains and consider only the carbon and nitrogen atoms (or Cα atoms) in the main chain, which are treated as points in 3D space. Only the geometry of structures is considered and details such as the identity of specific atoms are ignored. Thus, each protein structure is treated as a sequence of points. For each structure $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n)$, $|P|$ denotes the number of points in P, and $P_{i,j}$ denotes the fragment $(\boldsymbol{p}_i, \boldsymbol{p}_{i+1}, \cdots, \boldsymbol{p}_j)$ of P.

2.1 Root Mean Square Deviation

Here, we briefly review the root mean square deviation, which is used as a common measure for comparing two protein structures in molecular biology. The *rmsd* fitting is a kind of least-squares fitting method for two sequences of points, and was developed by several persons independently [5, 10, 13].

Let $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n)$ and $Q = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n)$ be two sequences of points. We assume that P is translated so that its centroid $(\frac{1}{n}\sum_{k=1}^{n} \boldsymbol{p}_k)$ is at the origin. We also assume that Q is translated in the same way. For each point or vector $\boldsymbol{x}$, $(\boldsymbol{x})_i$ $(i = 1, 2, 3)$ denotes the i-th (X,Y,Z) coordinate

value of $\boldsymbol{x}$, and $\|\boldsymbol{x}\|$ denotes the length of $\boldsymbol{x}$. Let

$$d(P, Q, R, \boldsymbol{a}) = \sqrt{\frac{1}{n}\sum_{k=1}^{n} \|R\boldsymbol{p}_k + \boldsymbol{a} - \boldsymbol{q}_k\|^2} \quad ,$$

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

$d(P, Q, R, \boldsymbol{a})$ is minimized when $\boldsymbol{a} = \mathbf{o}$ and $R = (A^t A)^{1/2} A^{-1}$, where the matrix $A = (A_{ij})$ $(i, j = 1, 2, 3)$ is given by $A_{ij} = \sum_{k=1}^{n} (\boldsymbol{p}_k)_i (\boldsymbol{q}_k)_j$, $A^{1/2} = B$ means $BB = A$, and $\mathbf{o}$ denotes the zero vector [13]. Thus, $d(P, Q)$, R and $\boldsymbol{a}$ can be computed in $O(n)$ time, where $O(f(n))$ time means that the computation time is at most $C \times f(n)$ for some constant C.

2.2 Substructure Search Problem

Using $rmsd$, we define the substructure search problem as follows (see Fig. 1):

Input: A (pattern) fragment $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_m)$, a real number $\delta > 0$ and a set of proteins $QS = \{Q^1, \cdots, Q^N\}$,

Output: All structures Q^j each of which contains at least a fragment $Q^j_{i,i+m-1}$ such that $d(P, Q^j_{i,i+m-1}) \leq \delta$.

The substructure search problem can be solved by a naive method which computes $rmsd$ for all $Q^j_{i,i+m-1}$'s. However, it takes $O(Nmn)$ time, where we assume that the length of each Q^j is $O(n)$. In fact, experimental results described in Section 4 show that it takes about a minute. It is too long for interactive uses of database management systems.

3 New Hash Vectors

3.1 Conditions for Hash Vectors

Before describing new hash vectors, we describe general conditions which should be satisfied by any hash vector. In conventional hashing methods, an integer number is associated with each object. However, in the hashing methods for protein structures, a vector of reals is associated with each fragment of fixed length. For each fragment $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_H)$ of length H, a hash vector $\boldsymbol{hs}(P)$ is associated. Then, the following conditions should be satisfied by $\boldsymbol{hs}(P)$:

(A) $\boldsymbol{hs}(P)$ is invariant with any isometric transformation (rotation/translation) for P,

(B) $\boldsymbol{hs}(P)$ is close to $\boldsymbol{hs}(Q)$ if $d(P, Q)$ is small.

Although condition (A) may be implied by condition (B), we describe them separately to make the presentation clear. Note that once such a vector is given, $d(P, Q)$ must be computed only when $\boldsymbol{hs}(P)$ is close to $\boldsymbol{hs}(Q)$. In the least-squares hashing method previously proposed [1], condition (A) is satisfied but condition (B) is not necessarily satisfied. Indeed, it sometimes fails to find similar fragments.

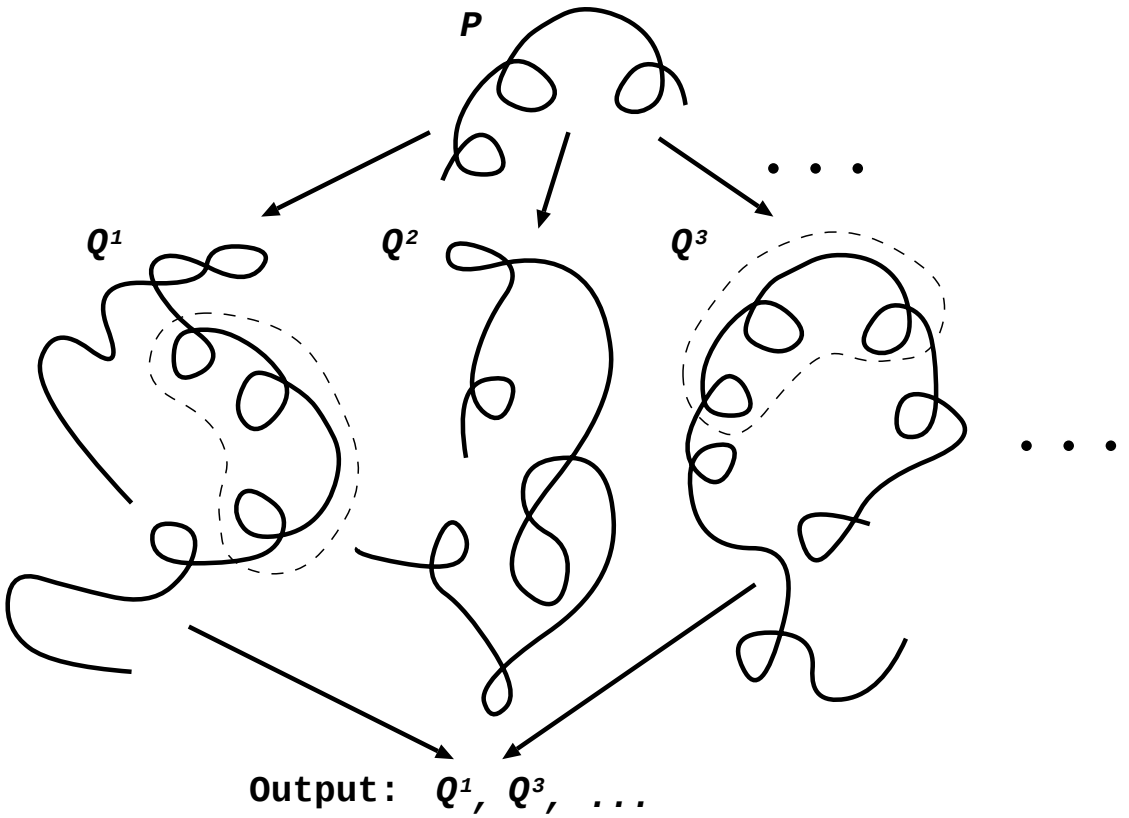

Figure 1: Substructure search problem.

3.2 Hash Vectors

Here, we describe new hash vectors. All vectors are very simple and computed in a similar way. First, we describe a basic one, denoted by HASH(A).

HASH(A):

$\boldsymbol{hs}(P) = (c_1(P), s_1(P), \cdots, c_D(P), s_D(P))$, where

$$c_i(P) \;=\; \alpha \sum_{k=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{c}\| \,(\cos(\frac{2\pi i(k-1)}{H}) + \beta) \;,$$

$$s_i(P) \;=\; \alpha \sum_{k=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{c}\| \,(\sin(\frac{2\pi i(k-1)}{H}) + \beta) \;.$$

Note that $\boldsymbol{c}$ denotes the centroid of P (i.e., $\boldsymbol{c} = \sum_{k=1}^{H} \boldsymbol{p}_k$). Also note that α ($\alpha > 0$) and β ($\beta \geq 0$) are fixed reals and D is a fixed integer, which are to be determined later. $\boldsymbol{hs}(P)$ is similar to (a low frequency part of) the Fourier spectrum of the distances between the points and the centroid (see Fig. 2). Although the Fourier spectrum has been already applied to geometric hashing in Ref. [6], $\boldsymbol{hs}(P)$ is quite different from it.

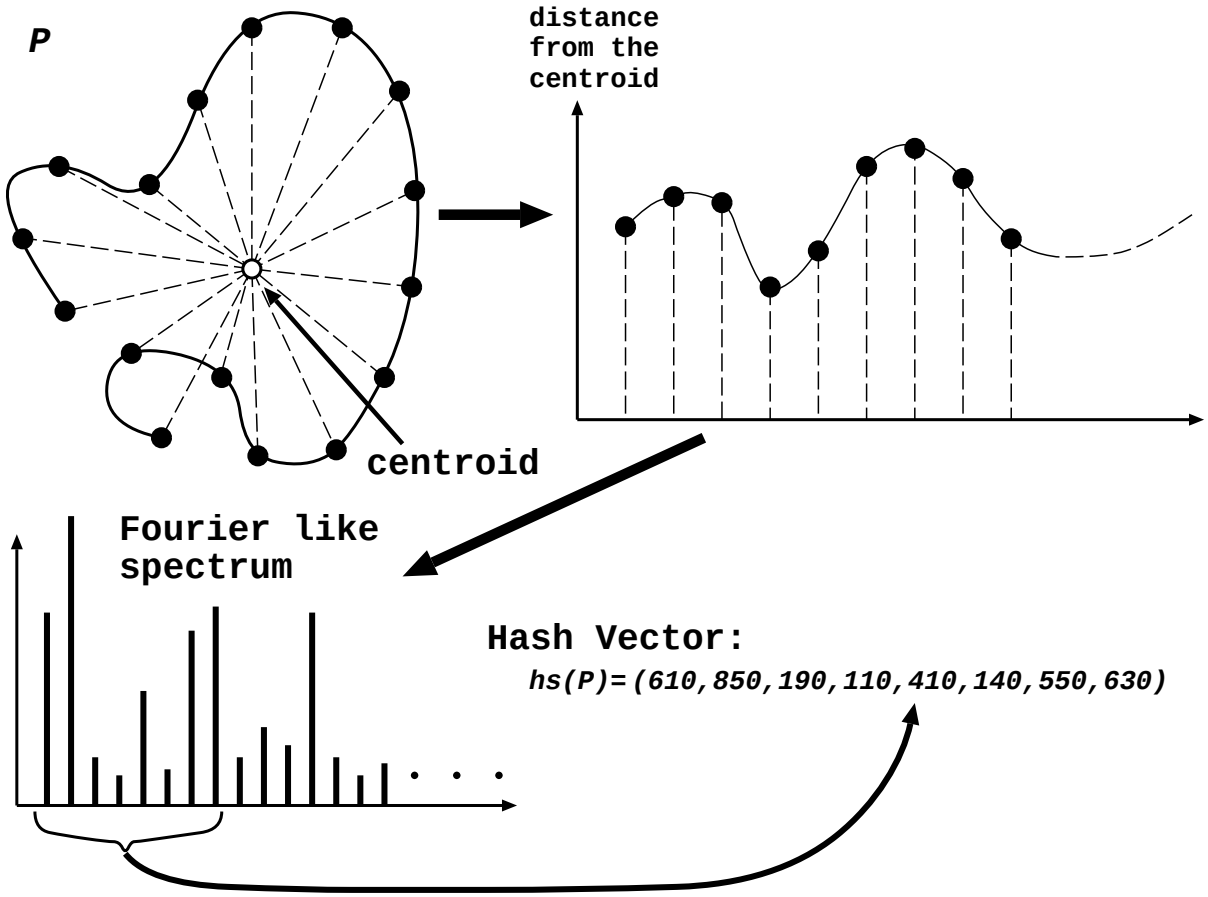

Figure 2: Computation of a hash vector.

For $\boldsymbol{hs}(P)$, condition (A) is trivially satisfied since $\boldsymbol{hs}(P)$ is computed only from the distances between the points and the centroid. To show that condition (B) is satisfied, we first prove the following lemma.

Lemma 1: Assume that $P = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_H)$ and $Q = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_H)$ are translated so that the centroids are at the origin. Then, $|\sum_{k=1}^{H} \|\boldsymbol{p}_k\| - \sum_{k=1}^{H} \|\boldsymbol{q}_k\|| \leq H\, d(P,Q)$ holds.

Proof: Let $\hat{Q} = (\hat{\boldsymbol{q}}_1, \cdots, \hat{\boldsymbol{q}}_H)$ denotes the rotated sequence of Q such that $d(P, \hat{Q}, I, \mathbf{o}) = d(P,Q)$, where I denotes the identity matrix.

Then, the following inequality holds:

$$|\sum_{k=1}^{H} \|\boldsymbol{p}_k\| - \sum_{k=1}^{H} \|\boldsymbol{q}_k\|| = |\sum_{k=1}^{H} \|\boldsymbol{p}_k\| - \sum_{k=1}^{H} \|\hat{\boldsymbol{q}}_k\|| \leq \sum_{k=1}^{H} |\,\|\boldsymbol{p}_k\| - \|\hat{\boldsymbol{q}}_k\|\,| \leq \sum_{k=1}^{H} \|\boldsymbol{p}_k - \hat{\boldsymbol{q}}_k\|,$$

where the last inequality comes from the triangular inequality. Since $t_1 + \cdots + t_H \leq \sqrt{H}\sqrt{t_1^2 + \cdots + t_H^2}$ holds for all $t_1 \geq 0, \cdots, t_H \geq 0$,

$$\sum_{k=1}^{H} \|\boldsymbol{p}_k - \hat{\boldsymbol{q}}_k\| \leq \sqrt{H}\sqrt{\sum_{k=1}^{H} \|\boldsymbol{p}_k - \hat{\boldsymbol{q}}_k\|^2} = H\, d(P,Q)$$

holds and the lemma follows. □

From Lemma 1, the following theorem is immediately proved, which shows that HASH(A) satisfies condition (B).

Theorem 1: For all i, $|c_i(P) - c_i(Q)| \leq H\alpha(1+\beta)d(P,Q)$ and $|s_i(P) - s_i(Q)| \leq H\alpha(1+\beta)d(P,Q)$ hold.

Let $COND(P,Q,\gamma)$ denote the condition that $(|s_i(P) - s_i(Q)| \leq \gamma \wedge |c_i(P) - c_i(Q)| \leq \gamma)$ holds for all i, where γ is a fixed real. From Theorem 1, the following property holds: if $COND(P,Q,\gamma)$ does not hold for $\gamma = H\alpha(1+\beta)\delta$, then $d(P,Q) > \delta$. Thus, if $COND(P,Q,\gamma)$ does not hold, it can be concluded that $d(P,Q) > \delta$ without computing $rmsd$ ($d(P,Q)$). However, note that $d(P,Q) \leq \delta$ does not necessarily hold if $COND(P,Q,\gamma)$ holds. Note that $\boldsymbol{hs}(P)$ can be computed in $O(H)$ time, and condition $COND(P,Q,\gamma)$ can be tested in constant time since we assume that D is a fixed small integer.

Next, we describe several variants of HASH(A). HASH(B) and HASH(B') are obtained by replacing the centroid $\boldsymbol{c}$ of HASH(A) with other positions.

HASH(B):
$\boldsymbol{hs}(P) = (c'_1(P), s'_1(P), \cdots, c'_D(P), s'_D(P))$, where $c'_i(P)$ (resp. $s'_i(P)$) is same as $c_i(P)$ (resp. $s_i(P)$) except that $\boldsymbol{d} = \sum_{k=1}^{L} \boldsymbol{p}_k$ is used in place of $\boldsymbol{c}$, where L is a fixed integer.

HASH(B'):
$\boldsymbol{hs}(P) = (c''_1(P), s''_1(P), \cdots, c''_D(P), s''_D(P))$, where $c''_i(P)$ (resp. $s''_i(P)$) is same as $c_i(P)$ (resp. $s_i(P)$) except that $\boldsymbol{e} = \sum_{k=N-L+1}^{N} \boldsymbol{p}_k$ is used in place of $\boldsymbol{c}$.

Next, we describe HASH(A+B) and HASH(A+B+B'), each of which is a combination of the vectors described above.

HASH(A+B):
$\boldsymbol{hs}(P) = (c_1(P), s_1(P), \cdots, c_D(P), s_D(P), c'_1(P), s'_1(P), \cdots, c'_D(P), s'_D(P))$.

HASH(A+B+B'):
$\boldsymbol{hs}(P) = (c_1(P), s_1(P), \cdots, c_D(P), s_D(P), c'_1(P), s'_1(P), \cdots, c'_D(P), s'_D(P),$
$c''_1(P), s''_1(P), \cdots, c''_D(P), s''_D(P))$.

Finally, we describe HASH(X). While all vectors described above correspond to a 1D Fourier spectrum, HASH(X) corresponds to a 2D Fourier spectrum. HASH(X) is similar to (a low frequency part of) a 2D Fourier spectrum of the distance map.

HASH(X):

$\boldsymbol{hs}(P) = (cc_{11}(P), cs_{11}(P), sc_{11}(P), ss_{11}(P),\ cc_{12}(P), cs_{12}(P), sc_{12}(P), ss_{12}(P), \cdots,$
$cc_{DD}(P), cs_{DD}(P), sc_{DD}(P), ss_{DD}(P)\ ),$ where

$$cc_{ij} = \alpha \sum_{k=1}^{H}\sum_{h=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{p}_h\| \left(\cos(\frac{2\pi i(k-1)}{H})\cos(\frac{2\pi j(h-1)}{H}) + \beta\right),$$

$$cs_{ij} = \alpha \sum_{k=1}^{H}\sum_{h=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{p}_h\| \left(\cos(\frac{2\pi i(k-1)}{H})\sin(\frac{2\pi j(h-1)}{H}) + \beta\right),$$

$$sc_{ij} = \alpha \sum_{k=1}^{H}\sum_{h=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{p}_h\| \left(\sin(\frac{2\pi i(k-1)}{H})\cos(\frac{2\pi j(h-1)}{H}) + \beta\right),$$

$$ss_{ij} = \alpha \sum_{k=1}^{H}\sum_{h=1}^{H} \|\boldsymbol{p}_k - \boldsymbol{p}_h\| \left(\sin(\frac{2\pi i(k-1)}{H})\sin(\frac{2\pi j(h-1)}{H}) + \beta\right).$$

It is not difficult to see that similar properties as Theorem 1 hold for all vectors except HASH(X). However, we have not yet proved a similar property for HASH(X).

3.3 Substructure Search Using Hash Vectors

Using $\boldsymbol{hs}(P)$, a substructure search can be done quickly in the following way. We assume that the length m of the fragment P is at least H. Moreover, we assume that $\boldsymbol{hs}(Q^j_{i,i+H-1})$'s are already stored along with all protein structures Q^j in a database. Although it takes $O(Hn)$ time to compute the hash vectors for each Q^j, the time can be ignored since the hash vectors for Q^j must be computed only when Q^j is stored in a database. The memory space required to store the hash vectors for each Q^j is $O(n)$.

To test whether or not there is a fragment $Q^j_{i,i+m-1}$ of Q^j such that $d(P, Q^j_{i,i+m-1}) \leq \delta$ holds, we use the following procedure.

STEP 1: For all $i \leq |Q^j| - m + 1$, test whether or not condition $COND(P_{1,H}, Q^j_{i,i+H-1}, \gamma)$ holds.

STEP 2: For all i satisfying the above condition, test whether or not $d(P, Q^j_{i,i+m-1}) \leq \delta$ holds.

Here, we consider the computation time for the above procedure. It is expected that only STEP 1 is executed for most i. Thus, the time for a protein structure Q^j is expected to be $O(n)$ in most cases. Thus, the time for N protein structures in a database is expected to be $O(Nn)$ in most cases.

Next, we consider the parameter γ. From Theorem 1, $\gamma = H\alpha(1+\beta)\delta$ should be used (in the case of $m = H$). However, experimental results show that it does not fail to find similar fragments even if a much smaller value is used. It is obvious that the search time is reduced if a smaller γ is used. Thus, the value for γ should be determined based on experimental results.

4 Experimental Results

Experiments have been done using PDB data [3]. Although PDB data contain various kinds of information, only positions of Cα atoms are used. All algorithms are implemented in the C language on a SUN SPARC STATION-10 (a UNIX workstation).

New hashing methods are compared with the previous and the naive ones in Table 1. NV denotes the naive method described in Subsection 2.2. LS denotes the least-squares hashing method [1]. Both A and A' denote HASH(A), where $D = 4$ is used in A and $D = 8$ is used in A'. B, A+B and A+B+B' denote HASH(B), HASH(A+B) and HASH(A+B+B'), respectively, where $D = 4$ and $L = \frac{H}{4}$ is used in each case. X denotes HASH(X), where $D = 3$ is used.

Each item in DATA denotes a filename (denoting a protein structure) of PDB data, where chain A is used in the case of 4HHB. Each pair of numbers in parentheses denotes the range of positions of a fragment P. Each item in NUM denotes the number of protein structures Q^j's each of which contains a fragment $Q^j_{i,i+m-1}$ such that $d(P, Q^j_{i,i+m-1}) \leq \delta$. For each algorithm and each pattern fragment, search time (CPU time (sec)) amongst all structures in a database is shown, where 811 structures are used and all structural data are stored in main memory of the workstation. A percentage of indices for which STEP 2 is executed is described too. In each algorithm except HASH(X), preprocessing (i.e., computation of hash vectors) for all structures was completed in a few minutes, so that it can be neglected. In HASH(X), it took more than an hour. However, it may be allowed since preprocessing must be done only once.

Table 1: Comparison of the new hashing methods.

DATA	NUM	NV	LS	A	A'	B	A+B	A+B +B'	X
4HHB(A) (50-94)	57	63.0	12.1 12.6 %	4.9 6.7 %	5.2 6.6 %	4.0 5.0 %	2.5 3.0 %	1.5 1.3 %	4.2 5.2 %
7LZM (35-80)	86	64.0	23.7 25.5 %	8.8 12.7 %	9.4 12.5 %	5.1 6.5 %	2.3 2.5 %	1.3 0.9 %	4.7 5.9 %
1R69 (5-55)	6	67.5	24.8 25.5 %	6.7 9.1 %	7.2 9.0 %	8.6 11.2 %	3.8 4.5 %	1.5 1.1 %	4.7 5.8 %
3ADK (115-170)	5	70.9	5.1 4.4 %	6.8 8.7 %	7.2 8.7 %	4.9 5.7 %	3.4 3.8 %	1.8 1.6 %	4.7 5.5 %
8LDH (150-194)	10	63.2	8.1 8.0 %	1.2 1.2 %	1.2 1.1 %	1.1 0.7 %	0.6 0.3 %	0.5 0.1 %	1.5 1.5 %

The following parameters were used: $H = 40$, $\alpha = 20.0$, $\beta = 0.5$, $\delta = 4.0(\mathring{A})$ and $\gamma = 1200.0$, where $\alpha = 1.0$ was used in HASH(X). In general, it is expected that search time is reduced if a larger value of D is used. However, comparison of A and A' shows that search time increases although a larger D is used, because the time for comparing hash vectors increases, while the percentage of indices for which STEP 2 is executed is reduced at most 0.2%. Thus, $D = 4$ is used. Although $\gamma \ll H\alpha(1+\beta)\delta$ was used, each method except LS could find all structures each of which contained a fragment $Q^j_{i,i+m-1}$ such that $d(P, Q^j_{i,i+m-1}) \leq \delta$. LS failed to find 3 structures in the case of 3ADK. Moreover, LS took longer than the other hashing methods in most cases. Thus, it is proved that new hashing methods are much more useful than the least-squares hashing method.

From Table 1, it is seen that the following relation holds approximately:

$$\text{A+B+B'} \succ \text{A+B} \succ \text{X} \approx \text{A} \approx \text{A'} \approx \text{B} \succ \text{LS} \succ \text{NV}\,,$$

where $x \succ y$ denotes that x is faster than y, and $x \approx y$ denotes that x is as fast as y. Thus, it can be said that we had better combine different types of hash vectors. This result seems natural from the following reason. In general, the position of a point in 3D can be determined uniquely except mirror image if the lengths from three fixed points are specified. Such vectors as HASH(A), HASH(B) and HASH(B') contain information about lengths from one point, while HASH(A+B) contains information from two points, and HASH(A+B+B') contains information from three points. Thus, combining different types of vectors, we can get more information about an outline shape of protein structure.

In each of the new hashing methods, it can be seen that search time was reduced considerably compared with the naive method. Especially, it is seen that HASH(A+B+B') is $30 \sim 100$ times faster than the naive method. Thus, it is proved that the new hashing methods, especially HASH(A+B+B'), are very effective.

5 Conclusion

In this paper, we have described new hashing methods for quick substructure searching. The experimental results show that the methods are very fast and effective.

Although the proposed methods work very well for small fragments ($|P| < 50 \sim 100$), they do not work well for large structures because they are not robust for insertions or deletions of sequences. Thus, an efficient hashing method, which is robust for insertions and deletions of sequences, should be developed.

Acknowledgement

This research was partially supported by the Grant-in-Aid for Scientific Research on Priority Areas, "Genome Informatics", of the Ministry of Education, Science and Culture of Japan.

References

[1] T. Akutsu, "Efficient and robust three-dimensional pattern matching algorithms using hashing and dynamic programming techniques," *Proc. 27th Hawaii International Conference on System Sciences*, Vol. 5, pp. 225–234. 1994.

[2] G. J. Barton and M. J. E. Sternberg, "LOPAL and SCAMP: techniques for the comparison and display of protein structures," *J. Molecular Graphics*, Vol. 6, pp. 190-196, 1986.

[3] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi and M. Tasumi, "The Protein Data Bank: A computer-based archival file for macromolecular structures," *J. Molecular Biology*, Vol. 112, pp. 535-542, 1976.

[4] C. Branden and J. Tooze, *Introduction to Protein Structure*, Garland Publishing, 1991.

[5] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta. Cryst.*, Vol. A32, pp. 922–923, 1976.

[6] A. Kalvin, E. Schonberg, J. T. Schwartz and M. Sharir, "Two-dimensional, model-based, boundary matching using footprints," *Int. J. Robotics Research*, Vol. 5, pp. 28-55, 1986.

[7] E. Kishon and H. Wolfson, "3-D curve matching," *Proc. AAAI Workshop on Spatial Reasoning and Multi-Sensor Fusion*, pp. 250-261, 1987.

[8] R. Nussinov and H. J. Wolfson, "Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques," *Proc. Natl. Acad. Sci. (USA)*, Vol. 88, pp. 10495-10499, 1991.

[9] C. A. Orengo and W. R. Taylor, "A rapid method of protein structure alignment," *J. Theoretical Biology*, Vol. 147, pp. 517-551, 1990.

[10] S. T. Rao and M. G. Rossmann, "Comparison of super-secondary structures in proteins," *J. Molecular Biology*, Vol. 76, pp. 241-256, 1973.

[11] M. G. Rossmann and P. Argos, "Exploring structural homology of proteins," *J. Molecular Biology*, Vol. 105, pp. 75-95, 1976.

[12] R. B. Russell and G. J. Barton, "Multiple protein sequence alignment from tertiary structure comparison: assignment of global and residue confidence levels," *PROTEINS: Structure, Function, and Genetics*, Vol. 14, pp. 309-323, 1992.

[13] J. T. Schwartz and M. Sharir, "Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves," *Int. J. Robotics Research*, Vol. 6, pp. 29-44, 1987.

[14] A. Tamura and M. Hirota, "A study on automatic methods for finding relationship between structural patterns and sequence patterns of proteins," Bachelor Thesis (in Japanese), Science University of Tokyo, 1994.

[15] W. R. Taylor and C. A. Orengo, "Protein structure alignment," *J. Molecular Biology*, Vol. 208, pp. 1-22, 1989.

[16] J. M. Thornton and S. P. Gardner, "Protein motifs and data-base searching," *Trends in Biochemical Science*, Vol. 14, pp. 300-304, 1989.

[17] G. Vriend and C. Sander, "Detection of common three-dimensional substructures in proteins," *PROTEINS: Structure, Function, and Genetics*, Vol. 11, pp. 52-58, 1991.