Clustering of Database Sequences for Fast Homology Search Using Upper Bounds on Alignment Score

Masumi Itoh itoh@kuicr.kyoto-u.ac.jp
Tatsuya Akutsu takutsu@kuicr.kyoto-u.ac.jp
Minoru Kanehisa kanehisa@kuicr.kyoto-u.ac.jp

Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan

Abstract

Homology data are among the most important information used to predict the functions of unknown proteins and thus fast and accurate methods are needed. In this paper, we propose a new approach for fast and accurate homology search using pre-computed all-against-all similarity scores in a target database. We previously developed a method for derivation of an upper bound of the Smith-Waterman score (SW-score) between a query and a homolog candidate sequence using the SW-score between the candidate and a sequence similar to the query. In this paper, by using this upper bound, we first cluster the sequences in the target database so that upper bounds of SW-scores for all the members in the clusters are less than a given value and select representative sequences for respective clusters. Then, the query sequence is searched against the representative sequences and the upper bounds of SW-scores for respective clusters are estimated. Only if the upper bound is higher than a given threshold, SW-alignments are computed for all the sequences in the cluster. We performed computational experiments to test efficiency of the proposed method for the KEGG/GENES database using the KEGG/SSDB. The results suggest that our method is efficient for redundant databases that include multiple closely related species.

Keywords: local alignment, homology search, dynamic programing, KEGG

1 Introduction

Recent rapid growth of sequence data enables large scale analysis of biological systems. A huge amount of gene and/or protein sequences is stored in many databases, but functions of a large part these proteins are unknown and it is almost impossible to experimentally analyze all of them. Thus computational methods to predict protein functions become important. For the prediction of protein functions, homology search is the most efficient and prevalent method. The Smith-Waterman algorithm [22] is known to be sensitive for detecting similarities between evolutionary related sequences, and the Smith-Waterman score (SW-score) is also used as a standard measure for the similarity between two sequences.

However, the SW-algorithm needs quadratic time for each pair of proteins and it takes a large amount of time to search for homologs in a large databases. To enable effective homology search against large databases, many methods have been proposed which improve calculation time by using heuristic methods, such as FASTA [14], BLAST [1], BLAST variants [2, 7, 24], and PatternHunter [15]. However, these methods never guarantee that they do not miss the sequences which have SW-scores higher than a given threshold. In the field of theoretical computer science, extensive studies have been done on nearest neighbor search [16] and fast sequence search [17, 18] based on the edit distance or similar distances. However, as shown in this paper, the SW-score is far from the distance and the results of such studies can not be applied.

On the other hand, there is another approach to reduce the computation time by storing all similarity scores among all pairs of homologous sequences in a database. Once we have constructed

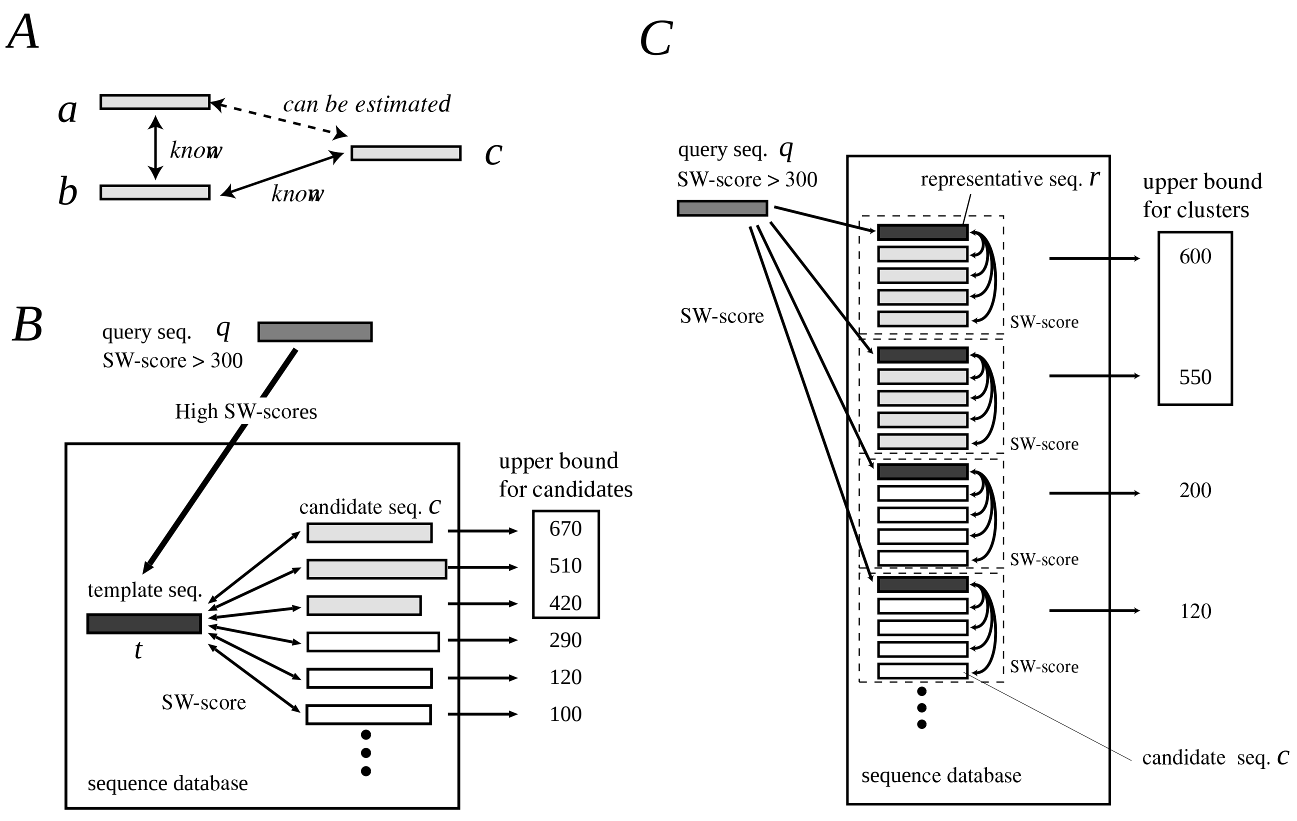

Figure 1: A: Estimation of upper bound of SW-score. If we know SW-alignment between a and b and $sw(b,c)$, we can estimate an upper bound of $sw(a,c)$. B: Fast and accurate search. In this case, SW-alignments are computed only for candidates whose upper bounds on $sw(q,c)$ are larger than the threshold (300). C: Clustering of sequences in database for fast search. By estimating upper bounds of SW-scores for all members, we can screen out the sequences whose SW-scores will never exceed the threshold (300).

such a database, we can obtain similarity scores for a query sequence against the database without score calculation if the query sequence is included in the database. This approach will be efficient for post-sequencing studies, because the query sequences will be already in the database and we only need to know their names. The KEGG/SSDB is one of such databases, it stores all homologous relations and gene clusters of ortholog candidates in the whole set of protein-coding genes in the KEGG/GENES [10, 11]. The KEGG/GENES contains gene/protein sets of completely and semi-completely sequenced species and their homologous relations are defined by the SW-scores.

However, this approach cannot be applied to sequences not included in the database. We previously developed a method for fast and accurate homology search using pre-computed SW-scores of all-against-all pairs of a target database [9]. The method is effective when the database includes at least one similar, but not identical, sequence to the query sequence. In this method, for three given sequences (a, b, and c), we focus on estimation of an upper bound of $sw(a,c)$ using $sw(b,c)$ and the SW-alignment between a and b (where $sw(x,y)$ denotes the SW-score between sequence x and sequence y). In other words, the possible space of $sw(a,c)$ is restricted by $sw(b,c)$ and the SW-alignment between a and b (Fig. 1A). If the similarity between a and b is high, the upper bound becomes more restrictive. Using this upper bound estimation together with a pre-computed SW-score database, we can screen out homolog candidates without computing SW-alignments. Since, for a given *query sequence* q and *threshold* θ, we can estimate whether the SW-scores against *candidate sequences* c can be higher than θ or not, without computing an SW-alignment when there is a *template sequence* t in the pre-computed database ($a \rightarrow q$, $b \rightarrow t$ and $c \rightarrow c$, respectively in Fig. 1A). Therefore, we only need to compute SW-alignments against candidates whose SW-scores may be higher than θ and the other candidates can be screened out. As described above, when q and t are very similar, the upper bound is very restrictive and a lot of candidates will be screened out, and we can drastically reduce the computation time for query sequences having similar sequences in the database (Fig.1B). This approach becomes very efficient when we carry out all-against-all comparison between the pre-computed database and the whole set of proteins from newly sequenced species, if closely related species are included in the database (from 6.7 to 50 times faster than complete SW-alignment). Though people in the artificial intelligence community applied the A* algorithm for computing alignments between long sequences by making use of some lower bounds [4], our proposed method and upper bound are quite different from such studies.

A conceivable third approach for fast search is pre-clustering of sequences in a database according to their homologous relations. Sequence databases usually include many similar sequences and it is a waste of resource to iterate calculation against all of them when the similarity with the query is low. Clustering similar sequences beforehand, we need to search only against sequences representing clusters to detect homologs. We can say that this pre-clustering approach is a method for reduction of redundancy in the database. In the ultimate case when the representative sequence and the other sequences included in the cluster are identical, we can simply merge these sequences into one and we can obtain the same result as in the non-clustering case. The efficiency of this approach will be improved by clustering not only identical sequences but also similar sequences. However, it is difficult to evaluate how high the similarities should be for merging the sequences into the same cluster. There are several previous works on clustering the sequences in a database to speed up searching and to improve sensitivity. Li *et al.* [12, 13] presented a method for fast and sensitive homology search by merging database sequences having high sequence identities. Dubey *et al.* [6] proposed a method for clustering proteins based on the theory of infinite Gaussian mixtures models. However, these methods did not have theoretical guarantees to obtain the same result as in the non-clustering case.

In this paper, we present a new criterion to cluster sequences and to select representative sequences of respective clusters without losing any sensitivity for homolog identification. This criterion is derived from the upper bound estimation of SW-scores, which we previously developed [9]. As described above, we can estimate an upper bound of $sw(q, c)$ from sequence similarity between ***q* and *t***. Using the same logic, we can also estimate an upper bound of $sw(q, c)$ from sequence similarity between ***t* and *c***. Therefore, we can estimate an upper bound of $sw(q, c)$ using $sw(r, c)$ and the SW-alignment between q and r where we assign the template sequence as a *representative sequence* r and a *candidate sequence* c as a member of the cluster. We can screen out entire clusters whose upper bounds are less than *threshold* θ if we can keep the upper bounds of $sw(q, c)$ less than a common value for all the members in a cluster. We just need to carry out the homology search against representative sequences and members of clusters whose upper bounds are higher than θ (Fig.1C). Then, we can reduce the computation time by avoiding comparison with redundant sequences in the database. Furthermore, because this approach for reduction of database size is independent of searching methods, any heuristic method can be combined with our method.

We performed computational experiments using the KEGG/GENES and the KEGG/SSDB. The results suggest that our method is very efficient for homology search against a redundant database. Recently, sequence redundancy tends to increase rapidly and our approach will be more efficient in the near future. Additionally, comparison of upper bounds and actual values indicates that we might improve efficiency of our method by estimation of tighter upper bounds.

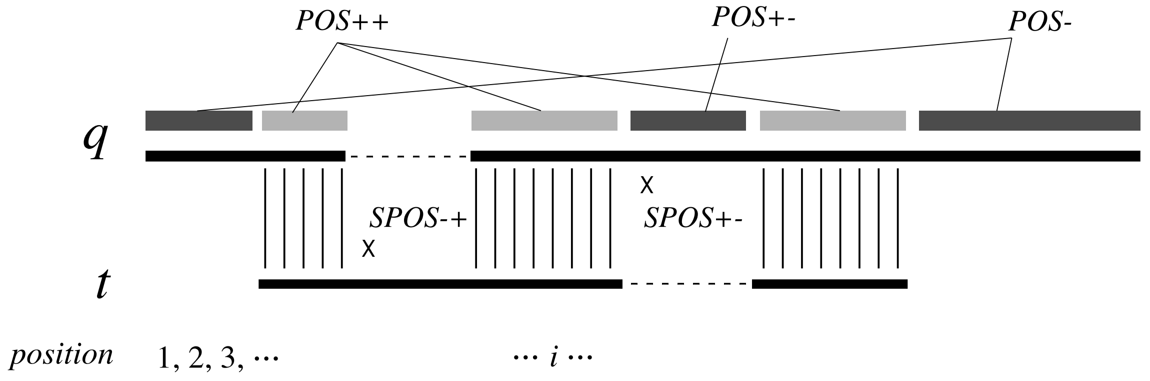

Figure 2: **Positions in local alignment.** q: query sequence, t: template sequence, POS_{++}: aligned positions between q and t, POS_{+-}: positions where gaps are inserted to t, $SPOS_{+-}$ and $SPOS_{-+}$: positions of opening-gaps (X in figure), POS_{-}: positions of q not appearing in local alignment.

2 Estimation of Upper Bounds on SW-Scores

Here we briefly review our previous method [9] for estimation of upper bounds on SW-scores.

The *triangle inequality* is a well-known property for the *distances*. If there were an appropriate distance measure $dist(x, y)$ between sequences x and y, we would have the following:

$$dist(q, c) \geq dist(q, t) - dist(t, c),$$

which could be used to estimate a lower bound of $dist(q, c)$ using $dist(q, t)$ and $dist(t, c)$ (a lower bound for the distance corresponds to an upper bound for the score). However, the SW-score is not a distance or does not satisfy the triangle inequality. So, in a previous work, we developed a method for deriving an upper bound of $sw(q, c)$ from (q, t) and $sw(t, c)$.

Let $\mathcal{A}$ and $s(x, y)$ denote the set of amino acids and a score matrix respectively (i.e., $|\mathcal{A}| = 20$, and $s(x, y)$ is a function from $\mathcal{A} \times \mathcal{A}$ to the set of reals). Let d and e be the cost per an opening gap and an extension gap, respectively ($d > 0$, $e > 0$). Suppose that (q', t') is a local alignment between q and t. We define POS_{++} (resp. POS_{+-}, POS_{-+}) to be the set of positions i such that q'_i is not a gap and t'_i is not a gap (resp. q'_i is not a gap and t'_i is a gap, q'_i is a gap and t'_i is not a gap). We define $SPOS_{+-}$ to be the set of positions i such that (q'_{i-1} is a gap or t'_{i-1} is not a gap) and (q'_i is not a gap and t'_i is a gap). $SPOS_{-+}$ is defined in an analogous way (see also Fig. 2). For $i \in SPOS_{+-} \cup SPOS_{-+}$, g_i denotes the length of the consecutive gap region starting from the position i. We define POS_{-} to be the set of positions of q that are not included in (q', t'). The following inequality holds for the SW-scores with afine gap costs:

$$\begin{aligned} sw(q, c) \quad \leq \quad & sw(t, c) + \sum_{i \in POS_{++}} \left[\max_{a \in \mathcal{A}} \{ s(q'_i, a) - s(t'_i, a) \} \right] \\ + \quad & \sum_{i \in SPOS_{-+} \cup SPOS_{+-}} \{ d + e(g_i - 1) \} \\ + \quad & \sum_{i \in POS_{+-}} \max_{a \in \mathcal{A}} \{ \ s(q'_i, a) \ \} \\ + \quad & \sum_{i \in POS_{-}} \max_{a \in \mathcal{A}} \{ \ s(q_i, a) \ \}. \end{aligned} \tag{1}$$

If the score matrix satisfies both of the following conditions:

$$\mathrm{argmax}_{a \in \mathcal{A}} \{ s(x, a) - s(y, a) \} \quad = \quad x$$

$$\text{argmax}_{a \in \mathcal{A}}\{s(x,a)\} \quad = \quad x$$

for all $x, y \in \mathcal{A}$, the inequality (1) can be transformed to:

$$sw(q,c) \;\le\; sw(t,c) + sw(q,q) - sw(q,t). \tag{2}$$

Most of the BLOSUM matrices [8] including BLOSUM50 satisfy these conditions. In this study, we use this inequality (2) because the KEGG/SSDB is constructed using BLOSUM50, where the same inequality was derived independently by Stojmirović [21]. However, our proposed method can be extended for other matrices such as PAM [5] using the inequality (1), which was not obtained by Stojmirović.

3 Clustering of Database Sequences

To screen out all the members of a cluster at once, the estimated upper bounds of SW-scores must be less than a given threshold for all its members. In this section, we will introduce a criterion to achieve such a requirement.

As described in the previous section, the upper bound of $sw(q,c)$ can be estimated from $sw(t,c)$ and SW-alignment between q and t, where q, c and t are *query*, *candidate* and *template* sequences, respectively. Especially for the BLOSUM50 matrices, the upper bound can be obtained from only $sw(t,c)$, $sw(q,c)$ and $sw(q,q)$. Since the inequality (2) is valid for any three sequences, we have

$$sw(q,c) \;\le\; sw(q,r) + sw(c,c) - sw(r,c),$$

where r is a *representative sequence* of a cluster and c is a member of the cluster. As shown here, the upper bound is derived as the sum of $sw(q,r)$ and $sw(c,c) - sw(r,c)$. We call the last part "*additive factor* $\sigma(r,c)$." When the upper bound of $sw(q,c)$ is less than *threshold* θ, we don't need to compute an SW-alignment between q and c. So the condition to screen out the candidates is

$$sw(q,r) + \sigma(r,c) < \theta.$$

Hence, if all members of a cluster satisfy this inequality, we can screen out the entire cluster at once. In other words, we need to cluster sequences such that all additive factors $\sigma(r,c)$ in a cluster are less than or equal to a given constant σ_{max}. To improve the efficiency of this process, we have to minimize the number of representative sequences. If σ_{max} becomes large, the sizes of the clusters also become large and the number of representative sequences becomes small. However, in such a case, estimated upper bounds of the clusters become large and only a few clusters will be screened out. So, there is a trade-off between σ_{max} and θ.

We can see that $\sigma(r,c)$ between two sequences is asymmetric. Difference between $\sigma(r,c)$ and $\sigma(c,r)$ becomes large when difference between the lengths of two sequences is very large, as in the case of multi-domain proteins. Such an example is shown in Figure 3. In the case of single domain protein a and its homolog a', $\sigma(a,a')$ and $\sigma(a',a)$ become very similar small values. When the two proteins are not similar such as a and g, both of $\sigma(a,g)$ and $\sigma(g,a)$ become large values. In contrast, if one sequence is a multi-domain protein such as ag, $\sigma(a,ag)$ and $\sigma(ag,a)$ are very different, $\sigma(a,ag)$ is very large but $\sigma(ag,a)$ is very small. Therefore, at the clustering process, ag can be a representative sequence for a cluster including a but a can not represent a cluster including ag. This asymmetric property shows the asymmetric relationship between multi-domain and single-domain proteins. When a query sequence is similar to g, we can not detect similarity between the query and ag if a represent a cluster including ag. But, if ag is a representative sequence for a cluster including a and g, the similarities to both sequences can be detected. In this case, g and a are included in the same cluster and we need to compute SW-alignments between g and a query protein similar to the protein a. However, searching

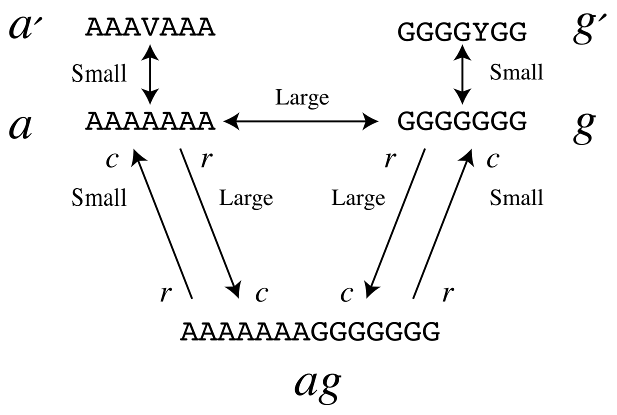

Figure 3: Asymmetricity of $\sigma(r, c)$ for two sequences.

time within the cluster is usually very small, compared with searching time against the whole target database.

When σ_{max} is fixed, we can deal with this problem as a variant of the *k-center problem.* It is difficult to solve the k-center problem and its variants in optimal way at large scale, thus we need to solve this problem using an approximate method. Here, we employed a greedy algorithm for large databases [19, 23]. The procedure is as follows:

1. Calculate $\sigma(x, y)$ for all pairs of two sequences x and y.
2. For each sequence x, count the number of sequences y (*neighbor sequences*) which satisfy $\sigma(x, y) < \sigma_{max}$.
3. Select a sequence with the maximum number of neighbor sequences as a representative sequence and put the neighbor sequences into a cluster represented by the selected sequence.
4. Remove the representative sequence and its neighbor sequences from the data set.
5. Iterate steps from 2 to 4 while there are remaining sequences.

The outline of this greedy algorithm are shown in Figure 4A.

4 Computational Experiments

4.1 Data Set

To estimate the practical efficiency of our clustering method, we applied it to the KEGG/GENES database using the KEGG/SSDB [10, 11]. This data set includes about 500,000 protein sequences from 132 species (159 strains in total). We obtained the whole set of SW-scores of all-against-all sequence comparison from the KEGG/SSDB and other sequence information was obtained from the KEGG/GENES. It should be noted that the KEGG/SSDB stores the pre-computed SW-scores (equal to or more than 70) among the whole set of proteins in the KEGG/GENES using the SSEARCH program [14] with default parameters (BLOSUM50 matrix, -12/-2 gap penalties).

Even if we employ the greedy algorithm for clustering sequences and selecting representatives, it is still difficult to cluster all the sequences of the KEGG/GENES at once. Hence, we used pre-clustered data from the KEGG Ortholog Clusters (OC). The KEGG Ortholog Clusters is the set

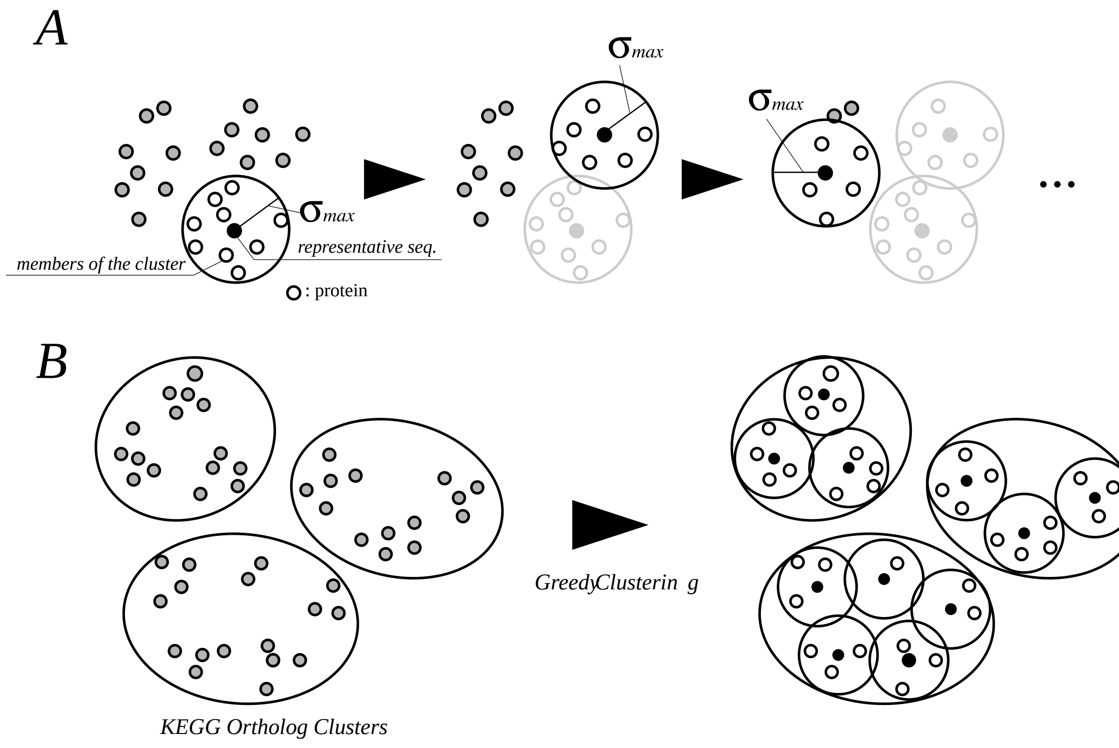

Figure 4: A: Outline of the greedy algorithm for clustering. B: Re-clustering of the KEGG Ortholog Clusters by the greedy algorithm.

of clusters of ortholog candidates in the KEGG/GENES. They are constructed based on the Bi-directional best hit relations stored in the KEGG/SSDB using a fully computational method. At the end of the building process of OC, a redividing process is invoked for improving granularity. Before the redividing procedure, it contains 114,492 clusters of which 24,706 clusters consist of two or more sequences and 19,220 clusters include sequences from different species. We clustered the sequences in Ortholog Clusters including two or more sequences using the greedy algorithm (Fig. 4B). Since the largest Ortholog Cluster has only 8,183 sequences, it can be divided into clusters in a short time by the greedy algorithm. Use of this pre-clustered data set is reasonable because sequences included in a cluster obtained by this algorithm should be in a Ortholog Cluster. We compared our method with the *simple merging method* in which only identical sequences were merged into one cluster and present the result in the next section.

4.2 Clustering for Large Scale Database

The matrices in Figure 5 show the efficiency of reduction between species in the KEGG/GENES. Each column and row correspond to each species in the KEGG/GENES. The order of species is arranged according to evolutionary relationship. Color (depth) of a cell in the matrices shows the number of pairs of sequences (from the species corresponding to row and column) included in the same cluster.

The upper right region is the result of our method and the lower left is that of the simple merging method. As shown here, there are few pairs in the lower left region, where most of them appear around the diagonal region. This indicates that the simple merging method could merge only sequences from several strains of the same species. In contrast, there are a lot of colored (shaded) cells in the upper right region, although most of them are small. Furthermore, the number of pairs between closely related species in the upper right region is significantly higher than that of the lower left region.

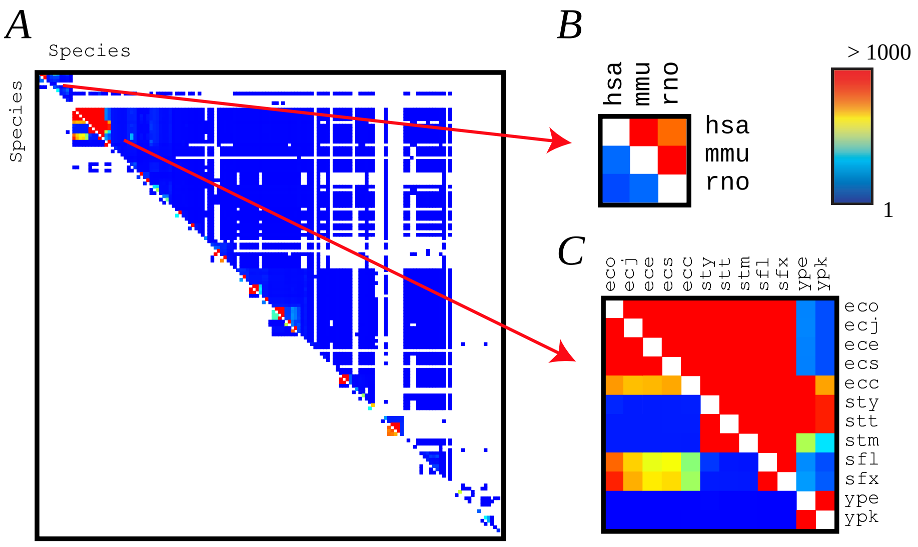

Figure 5: Number of sequences clustered into the same clusters from two different species. Colors show the numbers (see color scale). Rows and columns correspond to the respective species.

Table 1: Species closely related with *E. coli*.

Species	Strain	Abbrev.	#proteins
Escherichia coli	K12 MG1655	eco	4,289
	K12 W3110	ecj	4,390
	O157 EDL933	ece	5,347
	O157 Sakai	ecs	5,361
	CFT073	ecc	5,379
Salmonella typhi	CT18	sty	4,765
	Ty2	stt	4,323
Salmonella typhimurium	-	stm	4,553
Shigella flexneri	301	sfl	4,180
	2457T	sfx	4,948
Total	-	-	46,655

Figure 5B and C are close up views of such parts.

In total, our method is about 1.2 times faster than the naive method which computes SW-alignments between a query sequence and all sequences in a database. This is because the sequence redundancy is still not so high, but as mentioned above, the effect of our method in closely related species is very large. To test the efficiency for such closely related species, we applied our method to sequences from 10 genomes of proteobacteria included in the matrix of Figure 5C. We examined the clustering of sequences from respective species, and the clustering of all sequences. The species and number of their protein sequences in the KEGG/GENES are shown in Table 1. *Salmonella* and *S. flexneri* are closely related to *E. coli*, and they are often studied because of their pathogenicity.

Figure 6 shows the ratios of the representative sequences to the whole sequences. When we carry out homology search using our proposed method, we search against the representative sequences first, then we compute SW-alignments against the sequences included in the corresponding clusters if estimated upper bounds of SW-scores are higher than a given threshold. An upper bound of the SW-score is estimated as the sum of σ_{max} and SW-score against a representative sequence. Most of the sequences in a database usually don't have homology with the query sequences, and the number

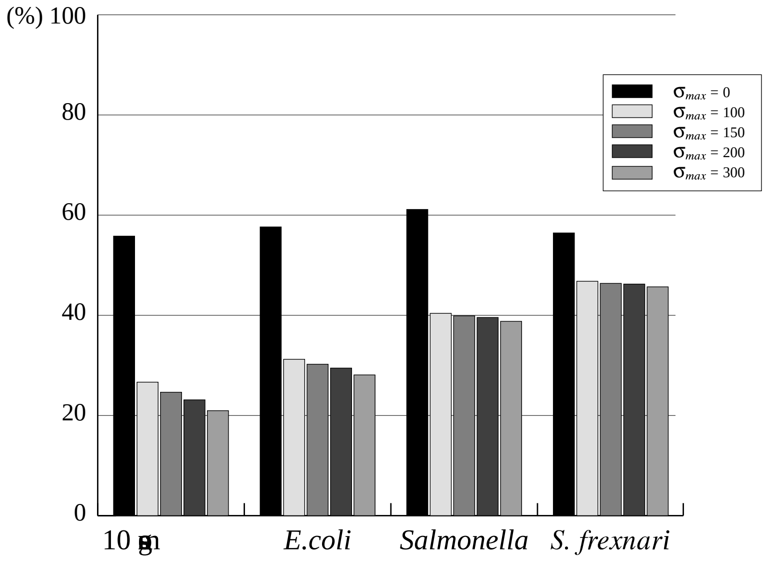

Figure 6: The ratio of the representative sequences to the whole protein sequences at respective σ_{max}. The numbers of representative sequences are equal to the numbers of clusters.

of sequences included in a cluster is vanishingly small, compared with the whole database size. Thus, we only need to consider computation time for SW-alignments against representative sequences, and the ratio of of the representative sequences to the whole database sequences directly shows the ratio of computation time of the proposed method to the naive method.
As shown in Figure 6, the ratios of representative sequences strikingly decrease between $\sigma_{max} = 0$ and $\sigma_{max} = 100$ at all groups of species. The case of $\sigma_{max} = 0$ in our method is equivalent to the case of the simple merging method. It suggests that a lot of sequences not merged by the simple merging method can be clustered by applying our method. The representative sequences were about 25% of the total number of sequences by clustering all of 10 *E. coli* related species at $\sigma_{max} = 150$. This means we can carry out homology search about 4 times faster for these species, and this is twice as efficient as the simple merging method. Clustering all sequences at once is more effective than clustering only sequences from one species, since redundancy of the sequences improves the efficiency of our proposed method.

5 Discussion

In this paper, we presented a new approach for fast homology search against large and redundant databases. The fundamental ideas of this approach are reduction of redundancy by clustering sequences and selection of representative sequences from respective clusters. With these, we can screen out most of the clusters according to SW-scores between the query and the representative sequences. We have introduced a clustering criterion to fasten homology search without losing sensitivity using estimation of upper bounds and pre-computed similarity databases.

The efficiency of this procedure was tested using the KEGG/GENES, the KEGG/SSDB and the KEGG Ortholog Clusters. Experimental results suggest that our proposed method will be very efficient for redundant databases. Analysis of species is biased due to human interest and genome projects for multiple strains of biologically important species, such as mammals, crops and pathogenic microorganisms, are ongoing. Therefore, the newly sequenced genes/proteins have a lot of redundancy and redundancy in the corresponding databases is increasing. Especially for human, mouse and rat (shown as hsa, mmu and rno in Fig. 5B, respectively), even when their genomes are not completely

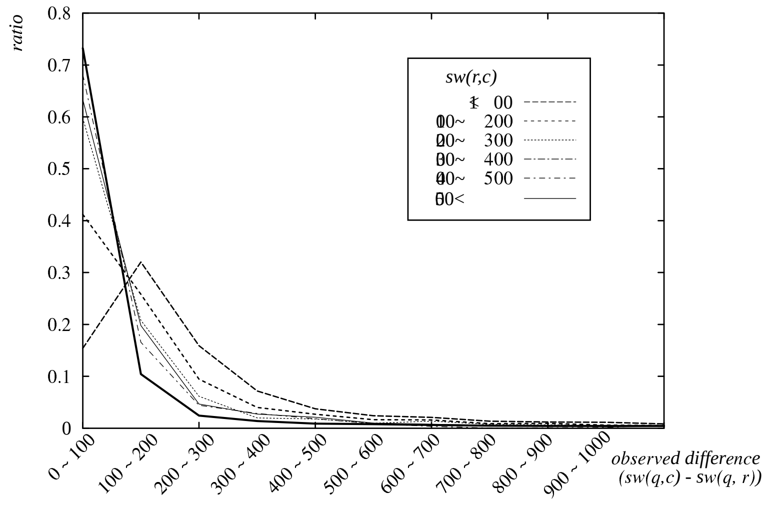

Figure 7: Distribution of actually observed differences at respective ranges of $sw(r, c)$. Vertical axis shows the ratio of sequence pairs of which observed difference are included in respective ranges.

sequenced and the KEGG/GENES includes only a part of their protein-coding genes (10% to 40%), many of their sequences are merged into clusters (Fig. 5), due to the fact that these mammals are closely related each other. Studies about mammals are one of the most important subjects in biology, and many genome sequencing projects are ongoing. This situation suggests that the proposed method will become more useful in the near future.

Essentially, the efficiency of our method is intrinsically related to the derived upper bound of SW-scores. Thus, the efficiency can be drastically improved if we can estimate the upper bound more tightly. By transforming the inequality (2), we can obtain the following inequality:

$$sw(q, c) - sw(q, r) \leq \sigma(r, c).$$

As shown here, the *additive factor* $\sigma(r, c)$ is an upper bound of the difference between $sw(q, c)$ and $sw(q, r)$. If upper bounds are estimated more tightly, $\sigma(r, c)$ will become smaller for all pairs of sequences and the average size of respective clusters will become larger, leading to faster homology search.

Practically, the additive factors were much higher than the actual difference. Figure 7 shows the distribution of actually observed differences, $sw(q, c) - sw(q, r)$, for respective ranges of $sw(r, c)$ under the condition that representative sequences r have homology with the query sequences (SW-scores $\geq$ 150). It is easy to estimate that $sw(r, c)$ is correlated with the difference between $sw(q, r)$ and $sw(q, c)$. When the representative sequence and the candidate sequence are not similar ($sw(r, c) \leq$ 100), most of the observed differences are larger than 100. On the other hand, when they have high similarity ($sw(r, c) \geq 500$), the observed difference are smaller than 100 in most cases (more than 70%, Fig.7). Additionally, when the representative sequence doesn't have homology with the query sequence ($sw(q, r) \leq 150$), the actually observed difference for almost all members of the clusters (98% at $sw(r, c) \geq 500$) are smaller than 100 (data not shown). This indicates that the upper bound could be derived more tightly using the information that *the representative sequence and the query sequence are not similar*. There are also several parameters to restrict the scores, such as length of sequences, ratio of overlap in whole sequences and amino acid contents. Therefore, we can improve the efficiency of our method by using this information to restrict the possible sequence space theoretically or statistically. We already developed a method to derive a tighter upper bound of SW-scores in a previous work [9], but the computation time to derive an upper bound is not small even for short sequences and so it is not a practical solution. We are still looking for more effective methods to estimate tighter upper bounds.

Application of the proposed approach is not limited to SW-search. It is easy to imagine that if a sequence is similar to one protein family, the sequence cannot be similar to other protein families which don't have any homologous relationship. Thus, the proposed method might be applicable to the family detection based on profiles of protein families such as HMM [3]. In our work, upper bounds were derived by considering the protein sequence space restricted by sequence similarities and alignments. Tighter restriction might be imposed by using other methods. To that end, we need to analyze the biological sequence space in more detail.

Acknowledgments

This work was supported by grants from the Ministry of Education, Culture, Sports, Science and Technology of Japan, the Japan Society for the Promotion of Science, and the Japan Science and Technology Corporation. The computational resource was provided by the Bioinformatics Center, Institute for Chemical Research, Kyoto University.

References

[1] Altschul, S.F., Gish, W., Miller, W., Myers, E., and Lipman, D.J., Basic local alignment search tool, *Journal of Molecular Biology*, 215:403–410, 1990.

[2] Altschul, S.F, Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J., Gapped BLAST and PSI-BLAST: A new generation of protein database search programs, *Nucleic Acids Research*, 25:3389–3402, 1997.

[3] Bateman, A., Coin, L., Durbin, R., Finn, R.D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E.L., Studholme, D.J., Yeats, C., and Eddy S.R., The Pfam protein families database, *Nucleic Acids Research*, 32:D138–D141, 2004.

[4] Davidson, A., A fast pruning algorithm for optimal sequence alignment, *Proc. 2nd IEEE International Symp. Bioinformatics and Bioengineering*, 49–56, 2001.

[5] Dayhoff, M.O., Schwartz, R.M., and Orcutt, B.C., A model of evolutionary change in proteins, *Atlas of Protein Sequence and Structure*, Vol. 5, Suppl. 3 (Dayhoff ed.), 345-352, 1978.

[6] Dubey, A., Hwang, S., and Rangel, C., Clustering protein sequence and structure space with infinite gaussian mixture models, *Pac. Symp. Biocomp.*, 9:399–410, 2004.

[7] Gish, W., WU-Blast 2.0. Website; `http://blast.wustl.edu`.

[8] Henikoff, S. and Henikoff, J., Amino acid substitution matrices from protein blocks, *Proc. Natl. Acad. Sci. USA*, 89:10915–10919, 1992.

[9] Itoh, M., Goto, S., Akutsu, T., and Kanehisa, M., Deriving upper bounds on local alignment score for fast and accurate homology search, *Manuscript*, 2004.

[10] Kanehisa, M., Goto, S., Kawashima, S., and Nakaya, A., The KEGG databases at GenomeNet, *Nucleic Acids Res.*, 30:42–46, 2002.

[11] Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., and Hattori, M., The KEGG resource for deciphering the genome, *Nucleic Acids Res.*, 32:D277–D280, 2004.

[12] Li, W., Jaroszewski, L., and Godzik, A., Clustering of highly homologous sequences to reduce the size of large protein databases, *Bioinformatics*, 17:282–283, 2001.

[13] Li, W., Jaroszewski, L., and Godzik, A., Sequence clustering strategies improve remote homology recognitions while reducing search times, *Protein Eng.*, 15:643–649, 2002.

[14] Lipman, D.J. and Pearson, W.R., Rapid and sensitive protein similarity searches, *Science*, 227:1435–1441, 1985.

[15] Ma, B., Tromp, J., and Li, M., PatternHunter: Faster and more sensitive homology search, *Bioinformatics*, 18:440–445, 2002.

[16] Muthukrishnan, S. and Sahinalp, S.C., Approximate nearest neighbors and sequence comparison with block operations, *Proc. 32nd Annual ACM Symp. Theory of Computing*, 416–424, 2000.

[17] Myers, E.W., An $O(ND)$ difference algorithm and its variations, *Algorithmica*, 1:251–266, 1986

[18] Myers, E.W., A sublinear algorithm for approximate keyword searching, *Algorithmica*, 12:345–374, 1994

[19] Panigrahy, R. and Vishwanathan, S., An $O(\log^* n)$ approximation algorithm for asymmetric p-center problem, *Journal of Algorithms*, 27:195–197, 1998.

[20] Rognes, T. and Seeberg, E., Six-fold speed-up of Smith-Waterman sequence database searches using parallel processing on common microprocessors, *Bioinformatics*, 16:699–706, 2000.

[21] Stojmirović, A., Quasi-metric spaces with measure, *ArXiv*, e-print:math.GN/0311070, 2003.

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

[23] Vazirani, V.V., *Approximation Algorithms*, Springer., 2001.

[24] Zhang, Z., Schwartz, S., Wagner, L., and Miller, W., A greedy algorithm for aligning DNA sequences, *Journal of Computational Biology*, 7:203–214, 2000