Parametric Alignment of Multiple Biological Sequences

Tetsuo Shibuya
shibuya@is.s.u-tokyo.ac.jp

Hiroshi Imai
imai@is.s.u-tokyo.ac.jp

[1] Department of Information Science, Faculty of Science,
University of Tokyo
7–3–1 Hongo, Bunkyo-ku, Tokyo 113, Japan

Abstract

The alignment problem of DNA or protein sequences is very applicable and important in various fields of molecular biology. In this problem, the obtained optimal solution with fixed parameters (gap penalties, weights for weighted multiple alignment problems, and so on) is not always the biologically best alignment. Thus, it is required to vary parameters and check the varying optimal alignments. The way to vary parameters has been studied well on the problem of only two sequences [6, 7, 12, 13, 14, 15], but not in the multiple alignment problem because of the difficulty of computing the optimal solution. This paper presents techniques for parametric multiple alignment problem, and examines the features of obtained alignments by parametric analysis on gap penalty and weight matrix through experiments. These experiments reveal the importance of adopting appropriate parameter values to obtain meaningful multiple alignments.

1 Introduction

The multiple alignment is a problem to obtain the alignment of multiple sequences with the highest score based on some given scoring criterion between characters. This problem appears in various fields of molecular biology such as the prediction of three dimensional structures of proteins and the inference of phylogenetic tree.

The method using dynamic programming (DP) is well-known for the alignment problems. This method needs $O(n^d)$ time and space for d sequences of length at most n. This method can be applied when n is not so large and d is 2 or 3, but for larger problems, it is impractical. The A* algorithm is a well-known algorithm for the general optimization and search problems. This algorithm can reduce the search space dramatically if a powerful estimator is used. Thus the A* algorithm with upper bounding operation is proposed recently for computing the optimal alignment of multiple sequences [8, 9].

In computing alignment, we set several parameters such as gap penalties, score matrices, and so on, based on our experiences through experiments. But, the parameter which induces the biologically best alignment is not always same in many cases. Hence we must check the solutions which are induced by various parameters.

The parametric 2-alignment problem has been studied very well [6, 7, 12, 13, 14, 15]. They did parametric analysis mainly on gap penalties. On the other hand, the parametric multiple alignment problem in respect to gap penalties is also an important and applicable problem, but has not been studied well yet mainly because of the computational difficulty of the problem. Furthermore, in multiple alignment problem, new parameters which do not appear in the 2-alignment problems are inherently introduced, and parametric problems for them should be also investigated. A typical example is the weighted multiple alignment problem, which is a generalized version of the simple sum-of-pairs multiple alignment problem [1, 4]. It has strong relationship to phylogenetic tree. This problem does not arise in 2-alignment problem, but is very important. In this weighted problem, we optimize sum of weighted pairwise scores, where the weights are expressed in matrices which we call weight matrices.

In this paper, we first show the (enhanced) A* algorithm is applicable for the weighted multiple alignment problem. We then review the techniques for parametric analysis, and propose new techniques for multiple alignments. As for the techniques, we introduce Eppstein algorithm to examine all the optimal solutions for one fixed parameter, and upper bounding technique for the parametric alignment. In most of previous works, they computed only one optimal solution for one fixed parameter in parametric analysis. We enumerate all the optimal solutions because the parametric analysis is the analysis of optimal solutions and we consider we should examine all optimal solutions. Fortunately, it is reasonable to obtain with Eppstein algorithm.

Then we present a parametric study on gap penalties using actual protein sequences. Furthermore, we also illustrate a parametric analysis on weight matrices. Weighted problem is considered only when the phylogenetic tree is given, but our approach enables more flexible study of the weighted multiple alignment problem. This problem is also studied using actual protein sequences. These experiments show the importance and usefulness of the parametric study in multiple alignment problem.

2 A* Algorithm for Weighted Multiple Alignment

The multiple alignment problem can be easily transformed to the shortest path problem on some grid-like directed acyclic graph with no negative edges. Let S_k be the kth sequence of d sequences to be aligned, and $n_k = O(n)$ be the length of S_k. Then suppose a directed acyclic graph $G = (V, E)$ such that $V = \{(x_1, \ldots, x_d) | x_i = 0, 1, \ldots, n_i\}$ and $E = \{(v, v+e) | v \in V, e \in [0,1]^d, e \neq \mathbf{0}\}$. In this graph, a path from $s = (0, \ldots, 0)$ to $t = (n_1, \ldots, n_d)$ corresponds to an alignment of the sequences.

In the alignment problem of two sequences, the length of an edge is defined from the score table between characters, and the length of a path from s to t equals the score of the corresponding alignment. Figure 1 shows an example of it. In the multiple alignment problem, the sum of all the scores for alignments of pairwise sequences is generally used as the score. Thus the score of the alignment equals the length of the corresponding path, defining length of each edge as the sum of the lengths of the corresponding edges in the graphs of pairwise

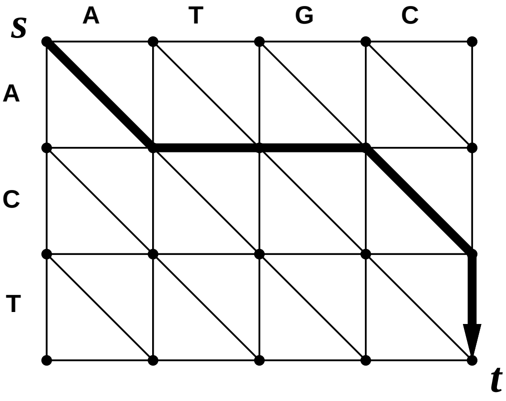

Figure 1: The graph for the alignment of two sequences ATGC and ACT. The s-t path in the bald line represents the alignment of ATGC- and A--CT

alignments. This longest path problem can be easily transformed to the shortest path problem by reversing the signs of the lengths [5, 8, 9]. From here, we discuss this transformed shortest path problem.

The A* algorithm will not search the whole graph in finding the shortest path if a good estimate for the shortest path length from each vertex to t can be used. Ikeda and Imai [9] show the following estimator is very useful in case $d > 2$. Let G_{ij} be the corresponding graph to the alignment of S_i and S_j, v_{ij} be the corresponding vertex in G_{ij} to v in G, and $L^*(u, v)$ be the shortest path length from u to v. Then $h(v) = \sum_{1 \leq i < j \leq d} L^*(u_{ij}, v_{ij})$ can be used as a powerful estimator for the multiple alignment problem. This estimator is easily be shown to be dual feasible, *i.e.* $l(u, v) + h(v) \geq h(u)$. Hence the A* algorithm can be applied as following.

1. For each of i and j ($1 \leq i < j \leq d$), apply DP to graph G_{ij} from t_{ij} to calculate $L^*(v_{ij}, t_{ij})$ for each v_{ij} in V_{ij}.

2. Modify the length of edge (u, v) in G as follows, using $h(v)$ above, and compute the shortest path with Dijkstra method. Notice that this new edge length is non-negative.

$$l'(u, v) = l(u, v) + h(v) - h(u) \tag{1}$$

Note that the time and space used for the DP in the step 1 is negligible, if d is large. This A* algorithm can deal with aligning 5 to 6 normal sequences in reasonable time.

A vertex in the graph for the multiple alignment has $2^d - 1$ edges going out from it, and the A* algorithm examines all the descendant vertices and keeps in a heap the information about all of them. If an upper bound $L^+(s, t)$ for the s-t shortest path, which corresponds to the lower bound of the score of the alignment, is given, the necessary space for the heap can be reduced [8]: we can ignore w such that $L^*(s, v) + l(v, w) > L^+(s, t)$, when we examine the descendant vertices of v. If the necessary space for the heap is reduced, the computing time of the A* algorithm will be also reduced. This is called the enhanced A* algorithm. Note that the branch-and-bound techniques implemented in MSA program [5] is equivalent to this enhanced A* algorithm.

The weighted (sum-of-pairs) multiple alignment problem [1, 4] is a generalization of the simple multiple alignment problem described above. This version of the problem is often used when the phylogenetic tree is given. In this problem, we optimize sum of weighted scores of each pairwise sequence alignments: we multiply the score of the alignment of the ith and the jth sequence by w_{ij}. We call (w_{ij}) a weight matrix. Computing the optimal solution of this problem by the (enhanced) A* algorithm is rather easy: all we have to do is using $h(v) = \sum_{1 \leq i < j \leq d} w_{ij} \cdot L^*(u_{ij}, v_{ij})$ as the estimator.

3 Parametric Multiple Alignment

In this section, we describe the techniques for parametric analysis of multiple sequence alignment problem.

3.1 Basic Techniques

In this subsection, we describe basic methods to check how the optimal solution varies as the parameters such as gap penalties change. The easiest approach for this kind of problem is to change the parameter little by little and check the optimal solution, but we cannot know how little we should change the parameter. Recently the techniques for parametric analysis are developed [6, 7, 12, 13, 14, 15]. In those previous works, they also did parametric analysis which deal with more than one parameters, but algorithms for them are not so efficient as the one-parameter case and it will often be nonsense if the parameters are not related each other. Thus we deal with only one parameter at one time in this paper.

We consider the case in which the score of some alignment A_i is expressed with parameter p as follows:

$$s_i(p) = a(A_i) + b(A_i) \cdot p \tag{2}$$

Gap penalty satisfies this expression for example.

From here, we explain how to divide 1-parameter (1-dimensional) space to regions in which the optimal alignments are always same. Let a_i be $a(A_i)$ and b_i be $b(A_i)$. Let p_i and p_j be the values of the parameter which satisfies $p_i < p_j$ and has different optimal solutions. Let the alignment A_i be the alignment with smallest value of b among the optimal alignments at $p = p_i$ and A_j be the alignment with largest value of b among the optimal alignments at $p = p_j$. Then this two alignments A_i and A_j has the same score at $p = p_{ij} = -\dfrac{a_i - a_j}{b_i - b_j}$. If the optimal score at $p = p_{ij}$ equals to $s_i(p_{ij}) = s_j(p_{ij})$, there are only two regions between p_i and p_j. Otherwise, we can apply the same technique recursively (*i.e.* apply between p_i and p_{ij} and between p_{ij} and p_j) to obtain such division. Figure 2 shows an example of this procedure.

Letting n be the number of regions which we want to obtain, we only have to compute the optimal solutions $2n - 1$ times. Thus we can efficiently do parametric analysis in the case of one parameter.

3.2 Eppstein Algorithm

In the previous subsection, the alignments with the largest or smallest value of b among the optimal alignments at some fixed parameter are required. These can be easily obtained by

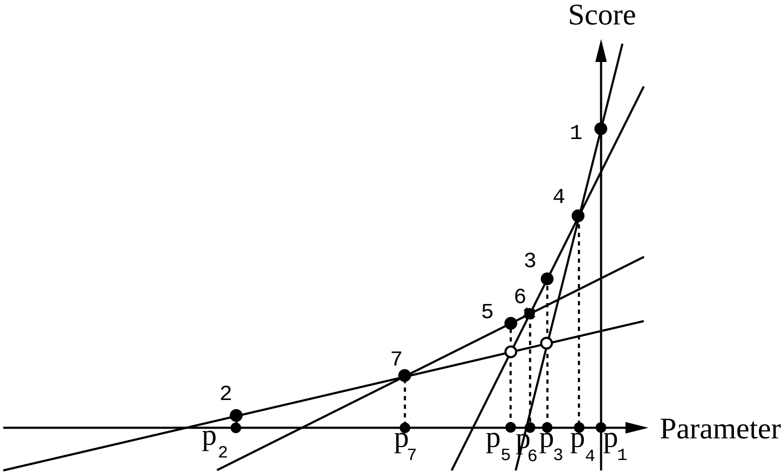

Figure 2: An example of division of 1-parameter space. In this case, there are 4 regions between p_1 and p_2.

some extension of DP [12, 13, 14, 15]. This technique can be applied also to the (enhanced) A* algorithm, but the aim of the parametric analysis is to examine all the optimal solutions. Accordingly, it is not preferable to ignore most optimal solutions if there are many. Fortunately, a new efficient algorithm for enumerating suboptimal solutions was recently proposed by Eppstein [3, 10, 11], which is known to be very useful for enumerating suboptimal solutions [10]. It is also efficient for enumeration of all the optimal solutions: the extra time of the enumeration is linear to the output size.

Let $\delta(u, v)$ for an edge (u, v) be $l(u, v) + L^*(s, u) - L^*(s, v)$. This $\delta(u, v)$ denotes how much longer the path will be using the edge (u, v) than the optimal path by way of v, and therefore this value is always non-negative. If an edge (u, v) is on the shortest path tree, $\delta(u, v)$ is zero, otherwise, it is called a sidetrack and $\delta(u, v)$ may not be zero. If we go along an s-t path p other than the shortest path, there must be sidetracks on the path, and we define $sidetrack(p)$ as the nearest sidetrack from s within them.

Let $(tail(p), head(p))$ be $sidetrack(p)$. Then we can suppose a heap, in which the parent of a path p is a path which is same as p from $head(p)$ to t, but go along the shortest path from s to $head(p)$ instead of using $sidetrack(p)$. We define this parent of p as $parent(p)$ and we call p a child of $parent(p)$. The root of the heap is the shortest path, and all the paths from s to t appear in the heap once. In this heap, p is $\delta(sidetrack(p))$ longer than $parent(p)$.

The basic concept of the Eppstein algorithm is constructing a graph which represents 4-heap modified from this path heap. From this heap, we can obtain the k shortest paths in $O(k)$ time, or $O(k \log k)$ time in sorted form.

3.3 Upper Bounding Technique for Parametric Alignment

As we stated in the section 2, the A* algorithm will be more efficient if some upper bounding value for the optimal solution is given (it is called the enhanced A* algorithm). In the parametric alignment problem, $s_i(p_{ij}) = s_j(p_{ij})$ in the subsection 3.1 can be used as this upper bounding value in computing the optimal alignments at $p = p_{ij}$.

4 Case Analysis and Experimental Results

In this section, we do parametric analysis of gap penalty and weight matrix. We also do experiments on actual protein sequence groups.

Concerning the score matrix, we used the famous PAM-250 matrix based on [2]. As for gap penalties, in the 2-alignment case, affine gap penalty is often used (*i.e.* penalty expressed as $a + bx$ where x is the length of the gap). On the other hand, in multiple alignment problem, to obtain the optimal alignment using the affine gap is very difficult, though there are many approximate algorithms which can deal with affine gap. Thus, we use linear gap penalty (*i.e.* penalty expressed as bx where x is the length of the gap) in this experiment. In the experiments for parametric weight matrix, we used -8 for gap penalty which is the minimum value in PAM-250 matrix. All the experiments in this section were done on Sun Ultra 1 workstation with 128 Mbyte memory.

We used 6 sequences of EF-1α sequences for experiments. This is a group whose similarity is very high and their lengths are about 430. Table 1 shows the sequences we used in the experiments.

4.1 Parametric Gap Penalty

We did parametric analysis of gap penalty using the top d sequences in Table 1.

In general, the most popular gap penalty is the minimum value in the score matrix, which is -8 in this PAM-250 case. We did parametric analysis for d-sequence alignment ($2 \leq d \leq 6$) with gap penalty between -2 and -16.

Table 2 shows the result of the experiment. In Table 2, the first row of each entry of d shows the boundaries of the regions, but several of the ends are not the boundaries: the ends with $-$ in #Max and #Min entry are not boundaries. The second row shows the number of the optimal solutions at the value. The last two rows shows the number of optimal solutions with largest/smallest value of b in the subsection 3.1. Thus, these values equal to the numbers of the optimal solutions between the boundaries.

According to the table, it is observed that the intervals of the regions become smaller as the penalty increases regardless of d. It also shows that there are not so much difference between different d's, which means we can do parametric analysis as easily as in the 2-alignment case. The table also shows that there are more than 1 optimal solution in all cases in the experiments.

Table 1: EF-1α sequences used for the experiments

Sequences			Pairwise Scores				
Species	Protein	Length	Met	Tha	Thc	Sul	Ent
Halobacterium marismortui (Hal)	EF-TU	421	1329	1314	1221	1109	1099
Methanococcus vannielii (Met)	EF-TU	428		1336	1247	1150	1176
Thermoplasma acidophilum (Tha)	EF-1α	424			1311	1261	1233
Thermococcus celer (Thc)	EF-1α	428				1132	1130
Sulfolobus acidocaldarius (Sul)	EF-1α	435					1192
Entamoeba histolytica (Ent)	EF-1α	430					

Table 2: The result of the experiment on parametric gap penalty.

$d = 2$	Gap penalty	-16	-5	-3	-2.5	-2				
	#Solutions	4	12	24	192	576				
	#Max	-	8	16	8	32				
	#Min	-	4	8	16	8				
$d = 3$	Gap penalty	-16	-3.5	-3	-2.75	-2.5	-2.2	-2		
	#Solutions	8	16	24	32	72	48	256		
	#Max	-	8	16	16	16	32	96		
	#Min	-	8	8	16	16	16	32		
$d = 4$	Gap penalty	-16	-8	-3.83	-3.5	-2.5	-2.33	-2.25	-2	
	#Solutions	16	32	32	32	32	48	160	4608	
	#Max	-	16	16	16	16	32	128	384	
	#Min	-	16	16	16	16	16	32	128	
$d = 5$	Gap penalty	-16	-7.5	-4	-3.38	-3.17	-3	-2.88	-2.75	-2.5
	#Solutions	2	4	4	4	4	4	12	8	24
	#Max	-	2	2	2	2	2	4	4	4
	#Min	-	2	2	2	2	2	2	4	4
$d = 6$	Gap penalty	-16	-6.5	-4.5	-4	-3.5				
	#Solutions	4	16	8	8	4				
	#Max	-	4	4	4	-				
	#Min	-	4	4	4	-				

Figure 3 shows the number of visited nodes by A* algorithm in computing the all the optimal alignments under various gap penalties. According to this figure, the number of the visited nodes increases drastically as the gap penalty increases especially when gap penalty is larger than -4. This is the reason why we analyzed gap penalty only up to -2.5 or -3.5 when $d \geq 5$: the required space was too large to compute when the gap penalty is around -2.

In general, the number of required space is large if the similarity among the group is low. This means that if the number of visited nodes becomes too large, similarity may not have been detected. Thus the gap penalty larger than -4 may be of no use.

4.2 Parametric Weight Matrix

Parametric analysis of weight matrix can be used for tuning parameters of a phylogenetic tree. A weight matrix for aligning sequences whose phylogenetic tree is known can be made if divergence between sequences are given [1]. But what should we do if the divergence are ambiguous? In such case, parametric analysis between reasonable two weight matrices helps.

There are $\frac{(d-2)(d+1)}{2}$ parameters to change in the weight matrix, thus what we can do is very limited simple analysis. We implemented a program to analyze how the optimal solutions change as weight matrix changes linearly between two weight matrices.

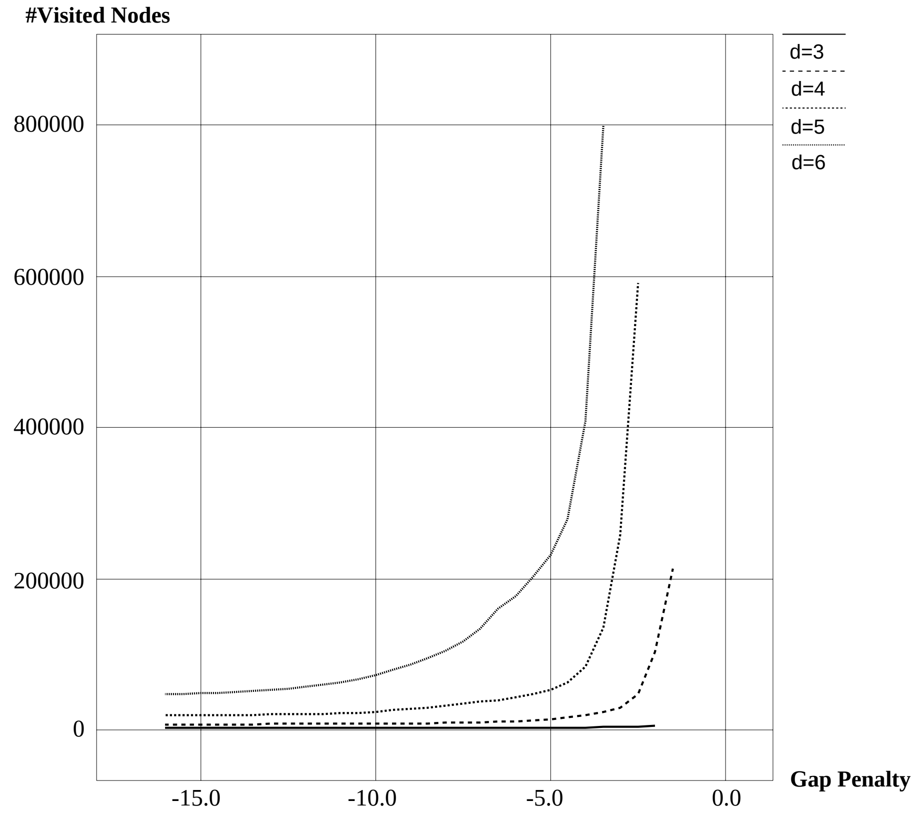

Figure 3: Number of visited nodes by A* algorithm under various gap penalties.

We did experiments between following two weight matrices of (w_{ij}) and $(w_{ij}^{(16,n)})$:

$$w_{ij} = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \qquad w_{ij}^{(p,n)} = \begin{cases} p \cdot w_{ij} & i = n \text{ or } j = n \\ w_{ij} & \text{otherwise} \end{cases} \tag{3}$$

In this equation, (w_{ij}) corresponds to the simple sum-of-pairs multiple alignment, and $w_{ij}^{(p,n)}$ increases the importance of nth sequence to p times as the simple sum-of-pairs multiple alignment. If biologically good alignment is discovered in the experiment, we can estimate the importance of the sequence which was increased.

Table 3 shows experiment results using the 6 EF-1α sequences in Table 1. The first column is the name of the sequence whose importance was increased. The first row of every entry shows the value of p of $w_{ij}^{(p,n)}$ which are boundaries of regions except for several ends with - in #Max and #Min entries. The second row shows the number of the optimal solutions and the other two rows shows the number of the optimal solutions with largest/smallest b in the section 3.1.

In this experiment, we notice that the optimal solutions will change even when only $p = 1.33$ in some of the cases (cases of **Tha** and **Ent**). It means we should take more care of the weight matrix. This experiment also show that there are more than 1 optimal solution in all the cases in this experiment. In the experiment, the number of the regions are not too large to deal with (6 to 10 in this experiment). This means this approach is very reasonable to take.

Table 3: The result of the experiment on parametric weight matrix.

Hal	Weight	1	1.33	3	3.17	4.5	4.75	5	6.57	16		
	#Solutions	4	16	16	12	8	8	48	16	8		
	#Max	-	4	8	4	4	4	8	8	-		
	#Min	-	4	4	8	4	4	4	8	-		
Met	Weight	1	2.25	3	4.2	4.4	4.5	12	14.5	16		
	#Solutions	4	8	12	16	16	16	16	16	24		
	#Max	-	4	8	8	8	8	8	8	16		
	#Min	-	4	4	8	8	8	8	8	8		
Tha	Weight	1	2.33	3	5.5	7.67	8	16				
	#Solutions	4	8	8	8	8	16	4				
	#Max	-	4	4	4	4	4	-				
	#Min	-	4	4	4	4	4	-				
Thc	Weight	1	1.8	3	4	5	5.33	6	10.33	12	14.44	16
	#Solutions	4	8	8	12	12	8	8	8	8	8	12
	#Max	-	4	4	2	4	4	4	4	4	4	8
	#Min	-	4	4	4	2	4	4	4	4	4	4
Sul	Weight	1	2	3	3.75	5	6	6.43	8	10.25	14	16
	#Solutions	4	12	8	8	12	16	16	16	16	16	8
	#Max	-	4	4	4	8	8	8	8	8	8	-
	#Min	-	4	4	4	4	8	8	8	8	8	-
Ent	Weight	1	1.33	1.5	3	3.66	5	6	10.33	13	16	
	#Solutions	4	8	8	8	8	8	8	12	24	16	
	#Max	-	4	4	4	4	4	4	8	16	-	
	#Min	-	4	4	4	4	4	4	4	8	-	

5 Concluding Remarks and Future Works

We introduced the concept of parametric analysis to multiple alignment problem. We also demonstrated parametric experiments on gap penalties and weight matrix for multiple alignment problem. Using parametric weight matrix technique practically in such problems as phylogenetic tree problems which are strongly related to weight matrix remains as one of future works. Applying similar techniques to other optimization problems in genome science, or doing parametric study of other parameters are also left as future works.

Acknowledgement

This work was supported in part by the Grant-in-Aid for Scientific Research on Priority Areas, "Genome Science" from the Ministry of Education, Science, Sports and Culture of Japan.

References

[1] S. F. Altsuchul, R. J. Carroll and D. J. Lipman, "Weights for Data Related by a Tree," *J. Mol. Biol.* 207, pp. 647-653, 1989.

[2] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *Atlas of Protein Sequence and Structure* (M. O. Dayhoff ed.), Vol.5, suppl. 3, pp. 345-352, National Biomedical Research Foundation, Washington D. C., 1978.

[3] D. Eppstein, "Finding the k Shortest Paths," *Proceedings of the 25th IEEE Annual Symposium on Foundation of Computer Science*, pp154-165, 1994.

[4] O. Gotoh, "A Weighting System and Algorithm for Aligning Many Phylogenetically Related Sequences," *CABIOS,* 11, pp. 543-551, 1995.

[5] S. K. Gupta, J. D. Kececioglu and A. A. Schaffer, "Improving the Practical Space and Time Efficiency of the Shortest-paths Approach to Sum-of-pairs Multiple Sequence Alignment," *Journal of Computational Biology*, Vol. 2, No. 3, pp. 459–472, 1995.

[6] D. Gusfield, K. Bakasubramanian and D. Naor, "Parametric Optimization of Sequence Alignment," *Proceedings of 3rd ACM-SIAM Annual Symposium on Discrete Algorithms,* pp. 432-439, 1992.

[7] X. Huang, P. A. Pevxner and W. Miller, "Parametric Recomputing in Alignment Graphs," *Proceedings of 5th Annual Symposium on Combinatorial Pattern Matching,* pp. 87-101, Springer-Verlag LNCS 807, 1994.

[8] T. Ikeda, "Applications of the A* Algorithm to Better Routes Finding and Multiple Sequence Alignment," *Master's Thesis,* Dept. of Info. Sci., Univ. of Tokyo, 1995.

[9] T. Ikeda, and H. Imai, "Fast A* Algorithms for Multiple Sequence Alignment," *Proceedings of Genome Informatics Workshop V*, pp.90-99, 1994.

[10] T. Shibuya and H. Imai, "Enumerating Suboptimal Alignments of Multiple Biological Sequences Efficiently," *Proceedings of Pacific Symposium on Biocomputing '97*, 1997, to be appear.

[11] T. Shibuya, T. Ikeda, H. Imai, S. Nishimura, H. Shimoura, and K. Tenmoku, "Finding a Realistic Detour by AI Search Techniques," *Proceedings of the 2nd Intelligent Tranportation Systems,* Vol. 4, pp. 2037-2044, 1995.

[12] M. Vingron and M. S. Waterman, "Sequence Alignment and Penalty Choices: Review of Concepts, Case Studies and Implications," *J. Mol. Biol.* 235, pp. 1-12, 1994.

[13] M. S. Waterman, "Introduction to Computational Biology: Maps, Sequences and Genomes," Chapman & Hall, 1995.

[14] M. S. Waterman, "Parametric and Ensemble Sequence Alignment Algorithms," *Bull. Math. Biol.* 56, pp. 743-767, 1994.

[15] M. S. Waterman, M. Eggert and E. Lander, "Parametric Sequence Comparisons," *Proc. Natl. Acad. Sci. USA,* 89, pp. 6090-6093, 1992.