Reprinted from Proceedings of

ICLP'95 Workshop on Parallel Logic Programming

Takashi Chikayama, Hiroshi Nakashima and Evan Tick (Eds.)

Published by Department of Electronic Engineering, University of Tokyo, Tokyo, 1995

This paper was presented on June 17, 1995, at Shonan Village Center, Hayama, Japan.

Application of a Parallel Logic Programming for Reconstruction of Molecular Phylogenetic Trees using the Maximum Likelihood Method

Satoshi OOta*, Naruya Saitou**, and Susumu Kunifuji*

* Graduate School of Information Science, Japan Advanced Institute of Science and Technology
15 Asahidai, Tatsunokuchi, Ishikawa, 923-12 Japan
** Laboratory of Evolutionary Genetics, National Institute of Genetics
Mishima, 411 Japan

Abstract

With rapid increase of DNA sequence data, it is required to develop reliable application programs to infer molecular phylogenetic trees in parallel environment. To implement a practical maximum likelihood method in parallel environment, we have developed programs *Contour/1* and *Traverse/3* to calculate likelihood values of arbitrary molecular phylogenetic trees. KLIC/KL1, a parallel logic programming language, was used for those programs. *Contour/1* generates log likelihood surfaces and we can investigate the surface to reach the maximum likelihood point, while *Traverse/3* searches the maximum likelihood values by traversing branches. We propose ideas how to implement the maximum likelihood method in a parallel environment.

1 Introduction

Reconstruction of molecular phylogenetic trees is quite important for molecular evolutionary studies. The molecular phylogenetic tree is one of the evolutionary models, which presents us how organisms evolved at the molecular level [8, 11].

The maximum likelihood method [3] is known to be relatively robust among many methods for reconstruction of molecular phylogenetic trees [6, 12]. In this method, the concept *likelihood* is defined as the measure for closeness between given data and a hypothesis. We explore the hypothesis space, and select one hypothesis which gives the maximum likelihood.

Unfortunately, however, this method requires extremely high computational cost [4]. One of practical resolutions for this problem is parallel execution. Program fastDNAml [9] is a maximum likelihood method implemented into parallel environment, however, it is expected that parallel logic programming provides us more appropriate concept for parallel execution for this problem. That is, the codes written with parallel logic programming is not only efficient in execution but also comprehensive for human.

ICOT (Institute for New Generation Computer Technology) developed an excellent parallel logic programming language, KL1. Using this language, we can easily write efficient codes for parallelism. KLIC also made it possible to implement programs written in KL1 to many different environments, from massively parallel computer to PC. It is expected that the maximum likelihood method in KLIC/KL1 leads us new discoveries through efficient and accurate data analysis.

2 Molecular Phylogeny

The molecular phylogenetic tree is the diagram which presents the evolutionary relationship of genes or organisms visually. There are two kinds of phylogenetic trees; species trees and gene trees [8]. In

this paper, we deal with the latter only.

There are three layers in a gene tree, the realized tree, expected tree, and estimated tree (Figure 1). We never know the expected tree precisely because everything we can infer is only mutations occurred on the branches.

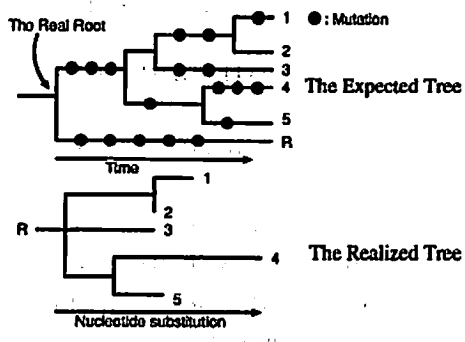


Figure 1: The realized tree and expected tree (modified from [11]).

The trees can be divided to rooted trees (see Figure 1) and unrooted trees (see Figure 5), but latter ones are usually produced in molecular phylogeny. If we specify the root, an unrooted tree will be transformed into a rooted tree.

The hypothesis space we have to explore is very huge. Even if we considered only the topology space, the number of its elements would increase immediately. The number of topologies of unrooted bifurcating trees for n nucleotide sequences is give as [8]

$$1 \times 3 \times 5 \times \cdots \times (2n-5) = \frac{(2n-5)!}{(n-3)!2^{n-3}}. \tag{1}$$

3 The Maximum Likelihood Method

The maximum likelihood method to infer a phylogenetic tree is based on the stochastic concepts. We assume trees which have certain branch lengths and topologies, and calculate the values of the likelihood on each tree, using the branch length, topology, and actual sequence data.

For instance, the likelihood (l) of the rooted tree below (Figure 2) for a particular nucleotide site in the sequence of K sites is

$$l = \sum_{s_5} \sum_{s_6} P_{s_6 s_1}(v_1) P_{s_5 s_2}(v_2) P_{s_6 s_3}(v_3) P_{s_6 s_4}(v_4) P_{s_0 s_5}(v_5) P_{s_0 s_6}(v_6) \pi_{s_0}, \tag{2}$$

where v_i $(i = 1 \cdots 5)$ is a branch length (number of nucleotide substitution), s_i is a state of character site of node i $(i = 1 \cdots 6)$, π_{s_0} is a prior probability of a nucleotide in a state s_0, and $P_{s_i s_j}(v_k)$ is the probability of a nucleotide transition from state s_i to s_j during the evolutionary time between nodes i and j.

We can generalize the above equation using the following equation [3]

$$l_{s_k}^{(k)} = [\sum_{s_i} P_{s_k s_i}(v_i) l_{s_i}^{(i)}][\sum_{s_j} P_{s_k s_j}(v_j) l_{s_j}^{(j)}], \tag{3}$$

where $l_{s_k}^{(k)}$ is the likelihood of node k with state s_k. $l_{s_k}^{(k)}$ is the product of likelihoods regarding two children nodes i and j (see Figure 3).

We assume the following Markov process.

$$\begin{cases} P_{s_i s_j}(dt) &= (1 - udt) + udt\pi_j & (i = j), \\ P_{s_i s_j}(dt) &= udt\pi_j & (i \neq j), \end{cases} \tag{4}$$

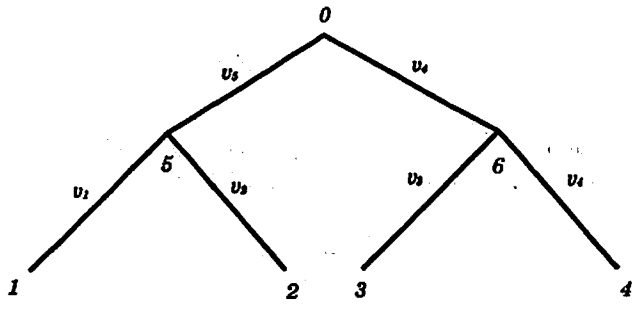


Figure 2: A tree for four sequences.

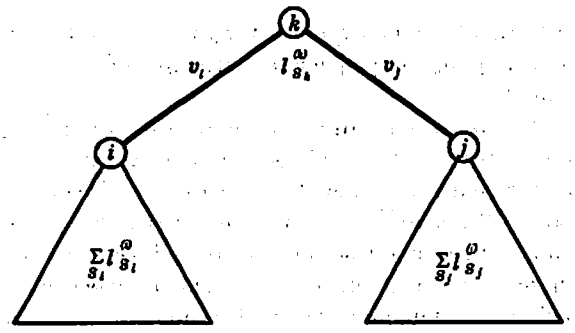


Figure 3: Recursive definition of conditional likelihood $l_{s_k}^{(k)}$.

where u is the substitution rate per unit time and t is the evolutionary time during nodes i and j [3]. Consequently, the probability that nucleotide s_i changes to s_j becomes

$$P_{s_i s_j}(v_j) = e^{-v_j}\delta_{s_i s_j} + (1 - e^{-v_j})\pi_{s_j}. \tag{5}$$

It should be noted that $v_i = u_i t$, which is the number of nucleotide substitutions at branch i accumulated during time t.

We explore all elements of the hypothesis space, and select the tree with the highest likelihood. In addition to the quite huge topology space, we have to search the branch length space in the given topology. Clearly, the likelihood of the given tree will change if branch lengths are modified.

To search for the stationary point of the likelihood surface, we calculate the differentiation of the likelihood with branch length, and find the point which makes the value of differentiation 0. Of course there is no guarantee that the stationary point gives the largest likelihood value. It is empirically known that the stationary point corresponds to the maximum value. But the strict theory of this calculation is remained to be established.

To derive the point which makes the differentiation 0, an iteration is applied. An appropriate initial value is set to the equation, and the derived value is set as a new initial value recursively.

We can calculate the conditional likelihood for any nodes using this relation. When we calculate conditional likelihood on the root, it means that we compute the calculation of the likelihood for the given tree. Since we usually do not know the state (nucleotide) of the root, we have to sum up the conditional likelihood with every possible state.

So far we have discussed about the evaluation of likelihood on the configuration of only one site. The actual likelihood is the product of the likelihoods through all sites of the sequences. Then the whole likelihood (L) can be presented as the following equation.

$$L = \Pi_k (A_k q + B_k p), \tag{6}$$

where $A_k = \sum_s \pi_s l_s^{(i)} l_s^{(j)}$, $B_k = \sum_{s_i} \pi_i l_{s_i}^{(i)} \sum_{s_j} \pi_j l_{s_j}^{(j)}$, $p = 1 - e^{-v_i}$, and $q = 1 - p = e^{-v_i}$.

We usually take logarithm of L, because calculation is simpler. Since logarithmic function is monotonic, we can evaluate the logarithm of L instead of L.

$$\ln L = \sum_k \ln(A_k q + B_k p). \tag{7}$$

We derive p instead of v_i, because we can obtain v_i from p easily.

4 Concurrent Algorithm

There are several ways of parallel computations in the maximum likelihood method. We should notice that our aim is not only to search topology space but also to calculate individual likelihood values of a given topology as fast as possible.

The most apparent computation which can be carried out in parallel is the topology search. Search of one topology is independent from that of another topology completely. However, the search space will increase exponentially if we execute exhaustive search. For instance, 20 sequence data produce the topology space whose size is 2.2×10^{20}. Thus it is necessary to specify candidates that might give the maximum likelihood value. We don't describe how to reduce the search space in this paper.

The evaluation for each site on the sequences can also be executed in parallel when we assume that mutations occur at each site independently. Furthermore, we can combine the sites which have same configurations, and compute at once.

The branch length searches are not independent, yet it is possible to execute each set of computations in parallel during the iteration since we apply iterative evaluation to obtain branch lengths. The maximum number of branches for n sequences is only $2n - 3$, and it is expected that this parallelism is efficient in limited parallel environment.

Actually, since the number of available nodes (processors) is strictly limited, we should assign appropriate priorities to *goals* according to data analyses we carry out.

5 Implementation

Our programs have been written in KLIC/KL1. At present, only *Contour/1* has been implemented to PIM/m and PIM/p, which are parallel inference machines developed by ICOT. In this section we show how to implement the algorithm to calculate conditional likelihood values and a sketch of the method to assign *goals* to available nodes.

5.1 Definition of a Tree

The way how to express a tree is critical in our case because a molecular phylogenetic tree is often unrooted. We cannot specify the root, which is the origin of the time stream.

Definition 1 *The structure of Tree is cons [Branch|Branches], where Branch is one of branches that compose the tree, and Branches is the rest of branches.*

Definition 2 *The structure of Branch is list (more generally we should call it functor) [$Node_1$, $Node_2$, Length], where $Node_1$ and $Node_2$ are two nodes of the branch, and Length is the length of branch flanked by both nodes.*

5.2 Definition of Conditional Likelihood

As mentioned above, the concept of conditional likelihood is applied recursively to calculate likelihood of the given tree. We can regard conditional likelihood as a probability that a node has a certain state.

Definition 3 *The predicate clike(Sites, Tree, Branch, State, Node, Node_Children, Likelihood) means that there is a node Node which has state State on one side of a branch Branch, and the node connects to two nodes Node_Children. These nodes and branches are on the tree Tree. The likelihood for that node is Likelihood.*

Therefore, if we specify *Sites*, *Tree*, *Branch*, *State*, *Node*, and *Node_Children*, conditional likelihood of a given node is obtained as *Likelihood*. The clauses which define predicate *clike/7* is as follows.

```
clike(Sites,Tree,Branch,State,Node,Node_Children,Likelihood)
        :- Node_Children \= [] |

        Node_Children = [Node_Child_1,Node_Child_2],

        States_1 = [a,t,g,c],
        get_time(Tree,Node,Node_Child_1,V_1),
        Next_Branch_1 = [Node,Node_Child_1],
        connect(Tree,Next_Branch_1,Node_Child_1,Node_Child_Children_1),
        sumPL(Sites,Tree,Next_Branch_1,States_1,State,V_1,Node_Child_1,
              Node_Child_Children_1,Left),

        States_2 = [a,t,g,c],
        get_time(Tree,Node,Node_Child_2,V_2),
        Next_Branch_2 = [Node,Node_Child_2],
        connect(Tree,Next_Branch_2,Node_Child_2,Node_Child_Children_2),
        sumPL(Sites,Tree,Branch,States_2,State,V_2,Node_Child_2,
              Node_Child_Children_2,Right),

        Likelihood $:= Left * Right.

clike(Sites,Tree,Branch,State,Node,Node_Children,Likelihood) :-
        Node_Children = []
        |
        site(Sites,Node,State_Node),
        commit_likelihood(State_Node,State,Likelihood).
```

Definition 4 *The predicate get_time(Tree, Node, Node_Child, V) means that the elapsed time, i.e., the branch length between two nodes Node and Node_Child is V. Of course Node and Node_Child are on the tree Tree.*

5.3 Direction of Evaluation

Although a node connects to three nodes except it is a leaf or a root, we have to suppress one of the directions of evaluation. A whole likelihood is a product of likelihoods of two subtrees and a transitional probability from a root of a subtree i to another root of another subtree j. To specify the tentative root of the subtree, we select a branch ij. This is the reason why the predicate *connect/4* requires *Branch* as an argument. A leaf can be the tentative root of a subtree, however, in this case, one of the subtrees is composed of only one node.

5.4 Parallel Execution

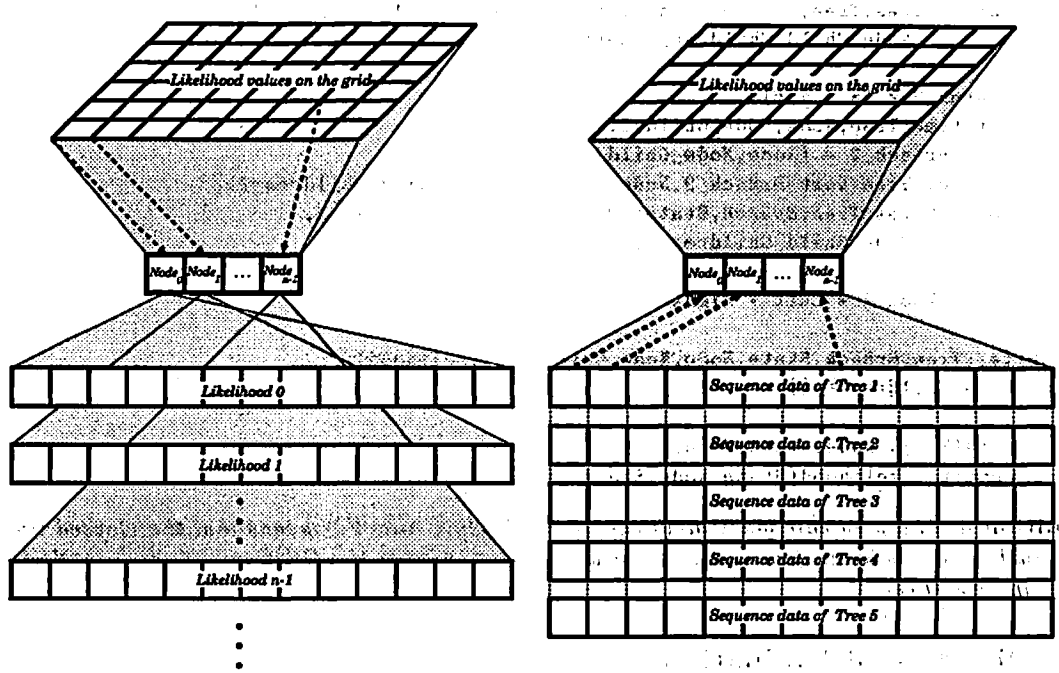
In *Contour/1*, each calculation of likelihood values is independent once branch lengths are given. Since the number of available nodes (processors) is less than points on a grid generally, each calculation

is distributed among them as equally as possible (Figure 4 (a)). The code to distribute is simple as follows (codes are simplified).

```
vary_x(Delta,X,Data,XYZss) :-
        X =< upper_limit_of_x
        |
        vary_y(Delta,X,Y,Data,XYZs),
        current_node(Current_Node,Total_Node),
        Next_Node := (Current_Node + Increment) mod 64,
        vary_x(Delta,Next_X,Data,Rest_XYZss)@node(Next_Node).

vary_y(Delta,X,Y,Data,XYZs) :-
        Y =< upper_limit_of_y
        |
        function(X,Y,Data,Z,XYZ),
        current_node(Current_Node,Total_Node),
        Next_Node := (Current_Node + 1) mod 64,
        vary_y(Delta,X,Next_Y,Data,rest_XYZs)@node(Next_Node).
```

Similarly, each computation for sites on sequences is assigned to nodes (processors) as many as possible (Figure 4 (b)). It is problem which of the calculations we should give priority whole likelihoods or individual sites (configurations). We decide the priority empirically in this program.



(a) Strategy 1 *(b) Strategy 2*

Figure 4: Two types of parallel execution.

6 Experimentation

6.1 Method

To examine the above condition, we have used the artificial data shown in (Table 1). There are

```
1       AAAAAAAAAAAAAAA CCCCCCCCCCCCCCC GGGGGGGGGGGGGGG TTTTTTTTTTTTTTT
2       AAAAAAAAAAAAAAA CCCCCCCCCCCCCCC GGGGGGGGGGGGGGG TTTTTTTTTTTTTTT
3       AAAAAAAAAAAAAAA CCCCCCCCCCCCCCC GGGGGGGGGGGGGGG TTTTTTTTTTTTTTT
4       AAAAAAAAAAAAAAA CCCCCCCCCCCCCCC GGGGGGGGGGGGGGG TTTTTTTTTTTTTTT
5       AAAAAAAAAAAAAAA CCCCCCCCCCCCCCC GGGGGGGGGGGGGGG TTTTTTTTTTTTTTT

1       AAACCGGTTT AACCGGT ACGTA CGT AC GT A CGTAC GTA C G
2       AAACCGGTTT AACCGGT ACGTA TAC GT GT A CGTAC GTA T G
3       AAACCGGTTT AACCGGT GTACG CGT GT GT C CGTAC CAG C A
4       AAACCGGTTT GTTGACC ACGTA CGT GT AC G TACGT CAG T T
5       GTGTGACACC AACCGGT ACGTA CGT GT CA T TACGT CAG T T
```

Table 1: Artificial nucleotide sequence data

60 invariant nucleotide sites in the five sequence data with 100 nucleotides. We have examined five topologies (see Figure 5). It may be noted that there are 15 possible topologies for five sequences. We have investigated the log likelihood surface around the inferred maximum likelihood point of the

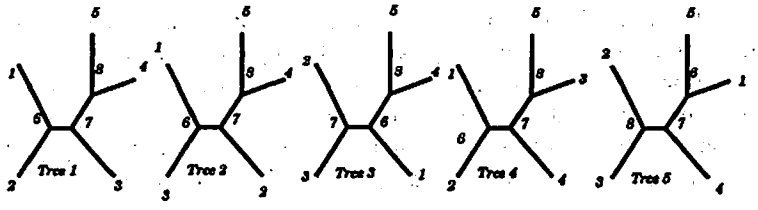


Figure 5: The examined five topologies.

given trees using *Contour/1*. DNAML [4] was first used to obtain the initial values.

Tree 1: [[1,6,0.02198],[6,7,0.02451],[7,8,0.11129],[3,6,0.08527],[2,7,0.04142],[4,8,0.09700],[5,8,0.15164]].

Tree 2: [[1,6,0.02746],[6,7,0.00006],[6,8,0.12823],[2,7,0.06089],[3,7,0.08712],[4,8,0.09427],[5,8,0.15429]].

Tree 3: [[1,6,0.02746],[6,7,0.00006],[6,8,0.12823],[2,7,0.06089],[3,7,0.08712],[4,8,0.09427],[5,8,0.15429]].

Tree 4: [[1,6,0.03428],[6,7,0.03189],[7,8,0.00003],[2,6,0.04899],[3,8,0.07872],[4,7,0.17928],[5,8,0.22394]].

Tree 5: [[1,6,0.05277],[6,7,0.00006],[7,8,0.00003],[2,8,0.06709],[3,8,0.08051],[4,7,0.19338],[5,6,0.23891]].

To present the log likelihood surfaces visually, lengths of user-specified two branches are varied and the corresponding log likelihood values are plotted. Lengths of other branches are fixed. In this experiment, two internal branches are specified to vary.

In parallel environment (PIM/m), we examined two kinds of parallel execution, which are calculations of total log likelihood values (Strategy 1) and calculations of individual likelihood values on each site (Strategy 2). The number of points on the grid is from 16(4×4) to 196(14×14), and the number of nodes (processors) is 64.

On the other hand, program *Traverse/3* searches the optimized branch length, and the predicate *branch/3* climbs the seven dimensional log likelihood surface to its peak. The predicate *branch/3* shows us the user time of its process using the predicate *times/4* in the module unix to reveal the performance [1]. In this manner, we evaluate optimized branch lengths of each tree, as well as the maximum likelihood.

Contour/1 was executed on PIM/m, while *Traverse/3* was executed on SUN SPARCstation CL™.

x\ y	0.00	0.01	0.02	0.03	0.04	0.05
0.15	-351.1923	-350.1430	-350.2229	-350.5290	-350.9350	-351.3973
0.16	-351.0913	-350.0823	-350.1648	-350.4710	-350.8765	-351.3378
0.17	-351.0291	**-350.0580**	-350.1431	-350.4493	-350.8541	-351.3146
0.18	-351.0013	-350.0661	-350.1537	-350.4599	-350.8640	-351.3234
0.19	-351.0043	-350.1032	-350.1931	-350.4993	-350.9026	-351.3610
0.20	-351.0350	-350.1662	-350.2584	-350.5645	-350.9671	-351.4244

Table 2: Log likelihood values around the inferred maximum likelihood point of Tree 2 (a boldfaced figure is the highest value)

6.2 Results

Some log likelihood values using *Contour/1* for Tree 2 are shown in Table 2, and its log likelihood surface is shown in Figure 2. In this example, lengths of two internal branches were varied from 0.00 to 0.30 and from 0.00 to 0.10 with the increment of 0.01. Table 3 shows the highest likelihood values which are inferred by using *Traverse/3* and corresponding inferred branch lengths for the examined five trees.

It is interesting that the curvature of the likelihood surface is asymmetric. When branches are short likelihoods change drastically, while the decrease of likelihood is not sensitive to branch length when branches are longer than the branch which gives the maximum likelihood value. This means that the likelihoods of the hypothesis space is not uniform, and improvement of likelihood depends on direction of search. In this case, we might overestimate lengths of these branches if we preset larger branch length than that which gives the maximum likelihood value. Actually, however, we happened to start from a smaller branch length and did not encounter such overestimation problem. It is useful to draw a likelihood surface not only for avoiding such misleading inference, but also for the efficient search of the hypothesis space. If a likelihood curvature is steep we can find the stationary point immediately.

Table 4 shows the number of reductions and the execution time of the two kinds of parallel execution (Test 1 and Test 2) using same the data. While the number of plots is less than 81, Test 2 is better than Test 1 in execution time, however, the situation changes when the number of plots is 196.

We show the change of likelihood of Tree 1 for each traverse of branches in Figure 7, which reveals that likelihood is improved drastically during the first traverse (in this case, *Traverse/3* evaluate seven branches at one traverse). However, the likelihood is not improved so much in the later ones.

The threshold, which is compared with improvement of log likelihood, to terminate program is set for 0.000001.

7 Discussion

7.1 Comparison with the results obtained by using the maximum parsimony method and DNAML

One of the most popular method to reconstruct molecular phylogenetic trees is the maximum parsimony method. Since computational cost is low and its concept is quite easy to understand intuitively, many evolutionalists have used this method not only in molecular level but also in phenotypic level. This method is suitable to examine our programs. Table 5 shows the application of the maximum parsimony method to the artificial sequence data of Table 1. There are two types of sites in the maximum parsimony concept; noninformative sites and informative sites. Only the latter type can be used to determine a topology. The definition of the informative site is that it has at least two different kinds of nucleotides, and each of them is represented in at least two sequences [8]. From Table 5 we can recognize which tree is most optimized in view of parsimony. Tree 1 is supported if we use the maximum parsimony method, since the total number of substitutions for this tree is the least. In

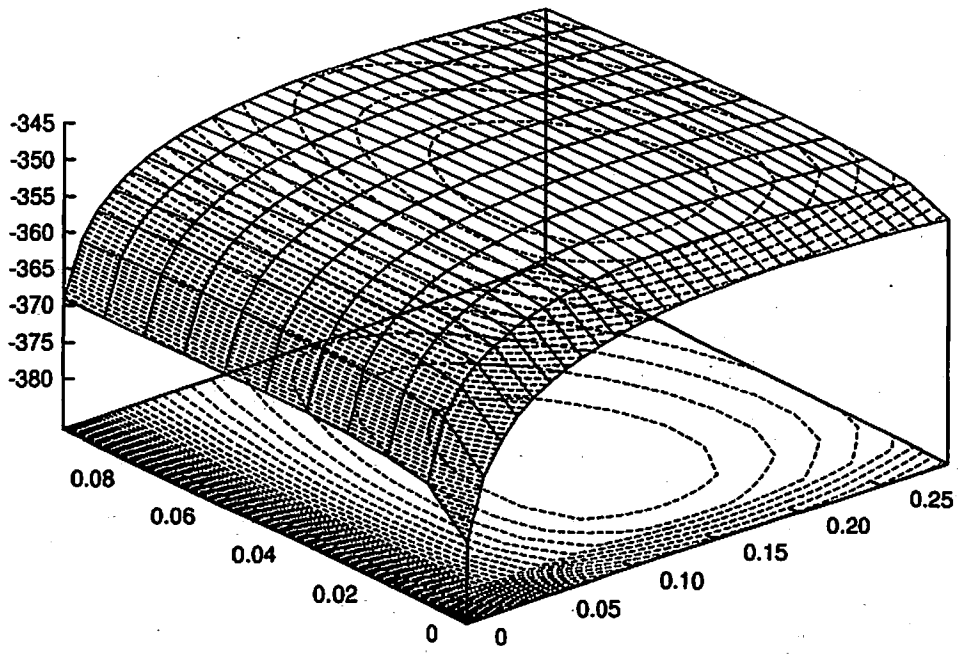


Figure 6: A log likelihood surface of Tree 2 using artificial sequence data of Table 2.

reality, this method may be positively misleading when the substitution rate is high [2]. In such case, we have to apply another method, for instance the maximum likelihood method.

In comparison with the results obtained by DNAML, since a different model of nucleotide substitution is used, its maximum likelihood values are slightly different from those of Table 3. However, the result of DNAML agrees well with ours as shown above tables. These mean that correctness of our program at least for the above artificial data.

7.2 Evaluation of Parallel Execution

Table 4 shows that the better strategy for parallel execution may change corresponding to size of data. As far as we investigated, there is no general distinction between two strategies in efficiency, however, the patterns of distributed load balancing are quite different. When the number of points on a grid is less than one of processors, the strategy of Test 2 tends to be superior to one of Test 1 in efficiency. In data analysis size of data may vary widely, so data analysts should make careful choices for better strategy. It should be noted that the execution time shown in Table 4 may change when *profile/1* [5] is not used.

7.3 Reduction of Search Space

Felsenstein [3] proposed the method to reduce topology search space. In his method, branches are added successively to the current maximum likelihood topology. Saitou [10] introduced an alternative strategy starting from the star phylogeny. Matsuda [7] proposed to assign lower priority to the topologies which give lower current likelihood, not to discard ones. Although these strategies do not

Tree 1: Maximum Log Likelihood = −344.525481

Estimated length	1–6	6–7	7–8	2–6	3–7	4–8	5–8
	0.031349	0.055529	0.124497	0.051341	0.096075	0.130399	0.185335

Tree 2: Maximum Log Likelihood = −348.403059

Estimated length	1–6	6–7	7–8	3–6	2–7	4–8	5–8
	0.029848	0.011697	0.167438	0.142440	0.044562	0.128639	0.185361

Tree 3: Maximum Log Likelihood = −349.078628

Estimated length	1–6	6–7	6–8	2–7	3–7	4–8	5–8
	0.034358	0.000001	0.175856	0.052834	0.145570	0.128258	0.185331

Tree 4: Maximum Log Likelihood = −352.774977

Estimated length	1–6	6–7	7–8	2–6	3–8	4–7	5–8
	0.039363	0.069882	0.000001	0.043210	0.114499	0.218438	0.269996

Tree 5: Maximum Log Likelihood = −362.553462

Estimated length	1–6	6–7	7–8	2–8	3–8	4–7	5–6
	0.065066	0.000001	0.000000	0.069616	0.128380	0.262359	0.315700

Table 3: Estimated branch lengths and the maximum likelihood of each tree

guarantee to reach the optimum tree, it is expected to alleviate the computation time, the great disadvantage of exhaustive search.

Another approach can be considered. It is natural to expect that more diverged sequences never cluster when other conserved sequences exist. We can ignore such inappropriate trees without computation of likelihood. The general strategy, which we will implement to PIM, is under consideration.

8 Conclusion

It is meaningful to draw a likelihood surface before inferring the maximum likelihood value. It may contribute to both accuracy and efficiency of the reconstruction of the true phylogenetic tree. To investigate how likelihood is improved is also important. Better strategy for parallel execution depends on data size critically. Also priority of goals in a same node is important not only for better performance but also for saving memory to proceed execution. We confirmed that parallel logic pro-

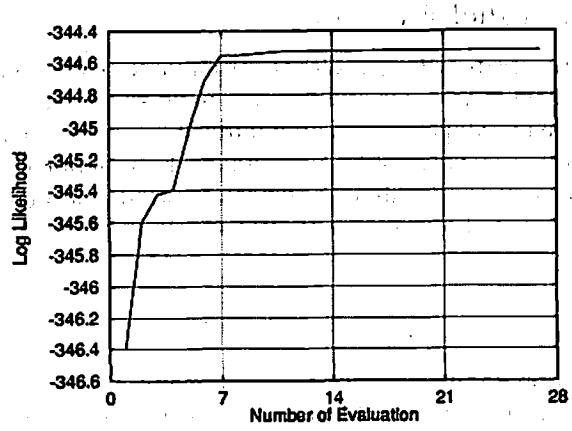


Figure 7: The change of likelihood of Tree 1.

Number of plots = 16

Calculation type	Number of reductions	Execution time
Strategy 1	26120144	44.4 sec.
Strategy 2	26101624	37.0 sec.

Number of plots = 25

Calculation type	Number of reductions	Execution time
Strategy 1	37583647	49.4 sec.
Strategy 2	37567280	47.3 sec.

Number of plots = 36

Calculation type	Number of reductions	Execution time
Strategy 1	51199188	58.6 sec.
Strategy 2	51100387	53.3 sec.

Number of plots = 81

Calculation type	Number of reductions	Execution time
Strategy 1	104412165	114.4 sec.
Strategy 2	104211904	103.2 sec.

Number of plots = 196

Calculation type	Number of reductions	Execution time
Strategy 1	234891074	206.5 sec.
Strategy 2	234393786	232.1 sec.

Strategy 1: Parallel execution for total log likelihood
Strategy 2: Parallel execution for individual likelihood on each site

Table 4: Two kinds of parallel execution with same data

gramming language KLIC/KL1 is quite useful to code maximum likelihood method in parallelism. We are going to develop this work mainly in the environment of PIM/p and PIM/m.

9 Acknowledgments

We appreciate kind advices of Dr. Kazuaki Rokusawa of OKI Electric Industry and Dr. Masato Ishikawa of Matsushita Electric Industry. We also thank the KLIC users group for providing us various informations on KLIC/KL1. We owe Eiji Sugino and Dr. Thanaruk Theeramunkong of Japan advanced Institute of Science and Technology for their technical advice of PIM.

References

[1] T. Chikayama. *KLIC User's Manual.* Institute for New Generation Computer Technology (ICOT), Tokyo, 1994.

[2] J. Felsenstein. Cases in which parsimony and compatibility methods will be positively misleasing. *Systematic Zoology*, 27:401–410, 1978.

[3] J. Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.

[4] J. Felsenstein. *PHYLIP: phylogeny inference package, ver. 3.5c.* University of Washington, Seattle, 1993.

[5] ICOT PIMOS Developing Group. *PIMOS Manual (V 3.0).* Institute for New Generation Computer Technology (ICOT), 1991.

	Sequence						Number of substitutions for tree				
i	A	B	C	D	E	m_i[a]	1	2	3	4	5
Noninformative configuration											
1	x	x	x	x	x	60	0	0	0	0	0
2	x	x	x	x	y	10	10	10	10	10	10
3	x	x	x	y	x	7	7	7	7	7	7
4	x	x	y	x	x	5	5	5	5	5	5
5	x	y	x	x	x	3	3	3	3	3	3
6	y	x	x	x	x	2	2	2	2	2	2
7	x	x	x	y	z	2	4	4	4	4	4
8	x	x	y	z	w	1	3	3	3	3	3
Informative configuration											
9	x	x	x	y	y	5	5	5	5	10	10
10	x	x	y	y	y	3	3	6	6	3	6
11	x	y	x	y	y	2	2	1	2	2	2
12	x	x	y	z	z	1	2	3	3	3	3
Total[b]						10	12	15	16	18	21

a) Observed number of configuration i.
b) Informative configurations only.

Table 5: The application of the maximum parsimony method to the artificial data

[6] M. Kuhner and J. Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular Biology and Evolution*, 11:459–468, 1994.

[7] H. Matsuda, S. Suzuka, and Y. Kaneda. A priority control mechanism for or-parallel prolog and its application (in Japanese). *Journal of Information Processing Society*, 34:773–781, 1993.

[8] M. Nei. *Molecular evolutionary genetics.* Columbia University Press, New York, 1987.

[9] G. J. Olsen, H. Matsuda, R. Hagstrom, and R. Overbeek. fastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. *Computer Application in the Bioscience*, 10:41–48, 1994.

[10] N. Saitou. Property and efficiency of the maximum likelihood method for molecular phylogeny. *Journal of Molecular Evolution*, 27:261–273, 1988.

[11] N. Saitou. Methods for building phylogenetic trees of genes and species. In H. G. Griffin and A. M. Griffin, editors, *Molecular biology: current innovations and future trends (in press).* Horizon Scientific Press, Norfolk, England, 1995.

[12] N. Saitou and T. Imanishi. Relative efficiencies of the Fitch-Margoliash, maximum parsimony, maximum likelihood, minimum-evolution, and neighbor-joining methods of phylogenetic tree reconstruction in obtaining the correct tree. *Molecular Biology and Evolution*, 6:514–524, 1989.