Evidence of Multiple Maximum Likelihood Points for a Phylogenetic Tree

B. B. Zhou[1], M. Tarawneh[1], P. Wang[1], D. Chu[1], C. Wang[1], A. Y. Zomaya[1], and R. P. Brent[2]

[1]*School of Information Technologies*
University of Sydney
NSW 2006, Australia
bbz@it.usyd.edu.au

[2]*Mathematical Science Institute*
Australian National University
Canberra, ACT 0200, Australia
rpb@rpbrent.co.uk

Abstract

An interesting and important, but largely ignored question associated with the ML method is whether there exists only a single maximum likelihood point for a given phylogenetic tree. Mike Steel presented a simple analytical result to argue that the ML point is not unique [11]. However, his view so far attracts only little attention. Though many researchers believe that multiple maximum likelihood points may exist for certain phylogenetic trees, most existing phylogenetic construction programs only produce a single best tree under the ML criterion and in practice many researchers still use only the ML values to make judgment on the quality of different trees for a given problem. In this paper we present some experimental results from a large number of synthetic test data sets and show that it is quite common that certain incorrect trees can have likelihood values at least as large as that of the correct tree. A significant implication of this is that even if we are able to find a truly globally optimal tree under the maximum likelihood criterion, this tree may not necessarily be the correct phylogenetic tree. In the paper we also show that our newly developed algorithm can perform much better in terms of accuracy than well known algorithms such as FASTDNAML and PHYML by constructing only a few more trees for a given problem.

1. Introduction

The maximum-likelihood (ML) method allows us to estimate and compare the likelihood of different evolutionary hypotheses (each comprising a model of the tree and substitution process), given the observed data [2]. It involves the definition of a labelled binary tree followed by optimisation of the estimates of length of edges on that tree. Experimental results often show that the ML method outperforms other methods, such as the maximum parsimony and the minimal evolution. As a result, the ML method is widely used in a variety of applications.

It is well known that one problem associated with the ML method is its high computational complexity. This has so far been considered as the biggest problem of the ML method. Most ML-based algorithms have therefore implemented heuristic local optimisation techniques to reduce the computational complexity, e.g., in the early days the simple stepwise addition proposed in [2] and more recently those described in [4,10,13].

An interesting and very important, but largely ignored question associated with the ML method is whether there exists only a single maximum likelihood point for a given phylogenetic tree. Though Mike Steel presented a simple analytical result to argue that the ML point is not unique [11], his view so far attracts only little attention. Till now the ML-based algorithm design focuses mainly on how to reduce the computational complexity and the designed algorithm is judged in terms of accuracy by how large the ML value an algorithm can produce for a given problem, and molecular biologists often use only ML values obtained from an ML-based algorithm to test phylogenetic hypotheses from genetic sequences. In this paper we present some experimental results from a large number of synthetic test data sets and show that it is quite common that certain incorrect trees can have likelihood values at least as large as that of the correct tree. A significant implication of this is that even if we are able to find a truly globally optimal tree under the maximum likelihood criterion, this tree may not necessarily be the correct phylogenetic tree! Consistency and low variance (when being compared with other methods) make the ML method a powerful approach of evaluating the uncertainty in phylogenetic estimates [5]. However, it is very important in practice to use different criteria and construct multiple trees for posterior analyses in order to obtain more accurate and reliable results.

To find an incorrect tree which has a likelihood value as large as that of the true phylogenetic tree can be equally hard as to find the true tree itself. This is simply because the search space is equally large. A two-phase procedure is adopted to solve the problem in our experiments. We first use a new quartet-based algorithm we recently developed [14] to construct a limited number of trees for a given set of DNA sequences. Our experimental results show that the probability for the correct tree to be included in this small set of trees is very high. Next in the second phase we calculate the likelihood values of these generated trees and choose just a few best ones which have the highest likelihood values. If there exists only a single maximum likelihood point for a given phylogenetic tree, we should be able to correctly identify the true phylogenetic tree if it is included in the set. Interestingly, our experimental results show that it is not always the case.

The paper is organized as follows: our quartet-based algorithm is briefly described in Section 2. In Section 3 we present some experimental results. Conclusions are given in Section 4.

2. The Quartet-Based Algorithm

In this section we briefly describe our quartet-based algorithm. The algorithm consists of two major stages. In stage one we calculate quartet weights for every possible quartet trees from a given number of molecular sequences. In stage two we first generate a global quartet weight matrix to gather the quartet topological information from the quartet weights calculated in stage one and then reconstruct a full size tree using this quartet weight matrix. We can use any existing method for phylogenetic analysis to calculate the weights of quartet trees [6]. In the following we only discuss the computations in stage two. We first discuss how a quartet weight matrix is generated from a set of quartets defined by a given tree topology and give an efficient algorithm for reconstruction of the tree topology from its generated quartet weight matrix. This one-to-one mapping between a given tree topology and its associated quartet weight matrix forms the basis of our quartet-based algorithm for phylogenetic analysis. We next discuss the tree reconstruction algorithm and show how to deal with quartet errors.

A quartet, or a set of four sequences is associated with three possible fully resolved trees. One way to measure which of the three possible trees is more likely to be the true tree is to use Bayes weights [6], or quartet weights in this paper. The quartet weights for three possible trees of a quartet is obtained by first calculating the likelihood value for each tree and then transforming these likelihood values into posterior probabilities, or quartet weights w_i for i = 1, 2, and 3, by applying Bayes' theorem assuming a uniform prior for all three possible trees.

Let $(ij \mid kl, w)$ denote a possible quartet tree with a quartet weight w. Our global quartet weight matrix is generated by adding each w to entries ij, ji, kl and lk, using a complete set of quartets from a given number of sequences. This matrix is symmetric and its size is $n \times n$, where n is the total number of sequences and each row or column corresponds to one particular sequence. (Note this quartet weight matrix is called score matrix in [3] and it was generated using discrete weights (or scores) from a distance matrix.)

Given a tree topology of n leaves, we can uniquely determine a set of quartet trees which are consistent with the original tree, i.e., each quartet tree separates the four leaves into two pairs in the same way as the original tree through bi-partitioning. For each quartet there can only be one fully resolved tree in the set of these quartet trees and it is described as $(ab \mid cd, 1.0)$. For a given tree topology a global quartet weight matrix is also uniquely determined. This is because our matrix is generated using the quartet weights of the associated quartet trees.

We can also reconstruct the tree topology from its generated quartet matrix by using an efficient $O(n^2)$ algorithm. In each step this algorithm tries to merge two subtrees using so-called confidence values which are obtained by comparing the corresponding matrix entry values with the desired ones. A detailed description of the algorithm can be found in [14].

The same algorithm described above may be used to reconstruct an evolutionary tree for a given set of n genetic sequences if all the associated quartets are fully and correctly resolved. Unfortunately, this is only an ideal case and in reality it is very hard for us to have all the quartets fully and correctly resolved. Therefore, the global weight matrix generated from a set of quartet weights is inaccurate and the algorithm for tree topology reconstruction discussed above cannot be used without modification. To deal with inaccurate weight matrices we make three major changes to the original algorithm.

The first change to the original algorithm is to use average confidence value. Since the entry values of the global weight matrix are no longer ideal, different node pairs, one from each of the two sub-trees, may produce different confidence values. A simple way to alleviate this problem is to calculate the confidence values for

every leaf node pairs, to average them and then to use this averaged value as the confidence value for each pair of sub-trees.

The second change is the quartet weight correction. After two sub-trees are merged, we take an additional step to restore the associated entries in the matrix to their "true" values, i.e., change the quartet weights based on the currently reconstructed sub-trees and update the weight matrix accordingly. In particular, after each merge we need to correct the weights of all those quartets containing four nodes {*i*, *j*, *p*, *q*} to (*ij*|*pq*, 1.0) where *i* is a leaf node in one merged sub-tree, *j* is a leaf node in the other merged sub-tree and *p* and *q* the leaf nodes from the rest. If the weights are not corrected, the distributed errors may significantly affect the correct decision making in the following merge steps.

The third major change is to construct multiple tree. Since the matrix is not accurate, it may not always be the right decision to merge the two sub-trees that have the highest confidence value. After the highest confidence value is obtained, we then check whether there is another sub-tree which has a reasonably high confidence value associated with one of the two sub-trees using a threshold which is a fixed number or a variable smaller than, but close to one. If it is the case, we keep all three different merge patterns of these three sub-trees. Therefore, we will reconstruct multiple trees and hope that the correct tree will be included in these generated trees. We set a limit on the number of trees to be reconstructed.

3. Experimental Results

We used a large number of synthetic data sets generated by the Methods and Algorithms in Bioinformatics Research group at The Montpellier Laboratory of Computer Science, Robotics, and Microelectronics in France (www.lirmm.fr). In our first experiment we used a total of 48,000 test data sets of DNA sequences. Six model trees, each consisting of 12 leaf nodes, are used to generate these test data sets under various situations. Three model trees, named AA, AB and BB, are molecular clock-like, while the other three, named CC, CD and DD, present varying substitution rates among lineages. Four evolutionary rates conditions are considered, ranging from low to very fast, for which the maximum pair-wise divergence (MD) is from 0.1 to about 2.0 substitutions per site. With these model trees and varying evolutionary conditions, the test data sets of DNA sequences, each being of length either 300 or 600, are generated using Seq-Gen. The detailed description of these synthetic test data sets can be found on their Web site at www.lirmm.fr. Some of our experimental results can be seen in Table 1.

In Table 1 QBNJ4+ML-*j* denotes that our algorithm QBNJ (using a fixed threshold $\alpha = 0.85$) with 4 stages (determining the number of maximum number of trees to be generated) is used to construct multiple trees and then *j* best trees are chosen under the ML criterion. In choosing *j* best trees we used relevant functions in a well-known programming package for phylogenetic analysis TREE-PUZZLE [12] to calculate the likelihood values of the generated trees and then these best trees are compared with the model tree using the Robinson and Foulds topological distance (RF) method [8]. The corresponding columns in the table show the percentages of correct trees among these *j* best trees for six different model trees. The figures are all rounded to their nearest integers. (In the table the figure x/y denotes the percentage of correctly inferred trees / the average number of trees generated per test data set.) The results obtained from FASTDNAML for the same test data set [7] are included in Table 1 for the purpose of comparison. We also ran one of the currently most well-known programming packages PhyML [4] on the same data sets and the results are presented in the table.

We can see from Table 1 that when only a single tree with the largest ML value is selected (QBNJ4+ML-1), the result is similar to (in some categories worse than) that obtained using FASTDNAML and PhyML. When allowing the selection of more than one tree, our QBNJ4+ML-*j* performs much better than both FASTDNAML and PhyML in almost all the categories.

In our second experiment we used the data sets with 24 sequences of length 600 based on 5,000 random 24 taxon trees, which are also downloaded from the web site www.lirmm.fr. These 5,000 trees are different in shapes and evolutionary rates. The internal branch lengths are also not all equal. In the experiment we used a variable threshold. Some experimental results are presented in Tables 2 and 3.

In Table 2 we compare the results produced by our algorithm using different numbers of stages with the result of PhyML. Using PhyML on these 24 sequence synthetic DNA test data sets, we can obtain 1385 (27.7%) correct trees in terms of topology. With only 3 stages allowed in our algorithm (QBNJ3), however, we are able to find 1760 (35.2%) correct trees by constructing on an average 25 trees for each test data set. Much better results are obtained when more trees are allowed to be generated.

After multiple trees were constructed for each test data set, similar to what we did in our first experiment

for 12 sequence data sets, we chose a few best trees under the ML criterion. Table 3 shows the number of correct trees we identified when choosing at most 5 best trees from those generated by our algorithm with 4 stages (QBNJ4) and 5 stages (QBNJ5). It is easily seen that both QBNJ4+ML-*j* and QBNJ5+ML-*j* performs better than PhyML when we are allowed to select just a few best trees.

Table 1. Experimental results of using QBNJ4 followed by ML for 12 sequence data sets, each sequence being of length 300.

		AA	BB	AB	CC	DD	CD
MD ≈ 0.1 :	QBNJ4	63/57	49/53	52/54	56/45	61/47	59/44
	FASTDNAML	23	20	21	16	18	18
	PhyML	22	22	23	16	18	18
	QBNJ4+ML-1	23	19	21	16	18	17
	QBNJ4+ML-2	32	26	28	25	26	25
	QBNJ4+ML-3	35	30	31	29	30	30
	QBNJ4+ML-4	40	32	33	32	32	33
	QBNJ4+ML-5	41	34	35	34	33	34
MD ≈ 0.3 :	QBNJ4	87/44	78/40	79/46	88/24	91/26	89/26
	FASTDNAML	58	53	54	70	68	69
	PhyML	58	56	57	67	67	70
	QBNJ4+ML-1	58	41	42	68	67	69
	QBNJ4+ML-2	69	64	65	78	78	77
	QBNJ4+ML-3	73	69	69	80	80	79
	QBNJ4+ML-4	75	70	71	80	80	79
	QBNJ4+ML-5	76	72	72	80	81	79
MD ≈ 1.0 :	QBNJ4	86/54	67/51	65/56	88/31	90/31	90/32
	FASTDNAML	44	36	37	82	83	81
	PhyML	40	31	34	73	75	75
	QBNJ4+ML-1	45	32	42	76	81	77
	QBNJ4+ML-2	45	42	44	82	88	83
	QBNJ4+ML-3	58	46	48	83	89	84
	QBNJ4+ML-4	61	49	49	83	89	85
	QBNJ4+ML-5	62	50	51	83	89	85
MD ≈ 2.0 :	QBNJ4	45/71	19/73	20/72	45/59	65/57	58/59
	FASTDNAML	8	4	5	59	62	59
	PhyML	5	3	3	26	28	28
	QBNJ4+ML-1	10	4	5	36	50	44
	QBNJ4+ML-2	14	6	8	40	56	49
	QBNJ4+ML-3	17	7	9	42	58	51
	QBNJ4+ML-4	18	8	9	42	58	51
	QBNJ4+ML-5	19	9	10	42	58	51

Table 2. Experimental results for 24 sequence data sets.

PhyML	QBNJ3	QBNJ4	QBNJ5	QBNJ7
27.7	35.2/25	47.5/61	56.5/159	63.1/371

Table 3. Experimental results of using QBNJ4 followed by ML for 24 sequence data sets.

PhyML	QBNJ4	QBNJ4+ ML-1	QBNJ4+ ML-2	QBNJ4+ ML-3	QBNJ4+ ML-4	QBNJ4+ ML-5
1385 (27.7)	2374 (47.5/61)	1017 (20.3)	1432 (28.6)	1695 (33.9)	1846 (36.9)	1957 (39.1)

PhyML	QBNJ5	QBNJ5+ ML-1	QBNJ5+ ML-2	QBNJ5+ ML-3	QBNJ5+ ML-4	QBNJ5+ ML-5
1385 (27.7)	2824 (56.5/169)	1095 (21.9)	1567 (31.3)	1877 (37.7)	2053 (41.1)	2192 (43.8)

We can see from both Table 1 and Table 3 that QBNJ4+ML-*j* (or QBNJ5+ML-*j*) is unable to identify all the correct trees from a limited number of trees generated by QBNJ4 (or QBNJ5). These are clear evidences that certain incorrect trees can have likelihood values at least as large as that of the correct one!

4. Conclusions

In this paper we presented some experimental results from a large number of synthetic test data sets to show that multiple maximum likelihood points do exist for a phylogenetic tree. This important result suggests that the ML criterion alone may not be sufficiently enough to determine the true phylogeny for certain problems even if we are able to obtain a truly globally optimal tree under the ML criterion! Therefore, it is very important in practice for us to use different criteria and construct multiple trees for posterior analyses in order to obtain more accurate and reliable results.

In the paper we also showed that our QBNJ+ML-*j* algorithm performs much better than FASTDNAML and PhyML for synthetic test data sets. It should be noted that our algorithm is computationally more expensive though it takes a polynomial time to complete. However, the accuracy is much more important than the computational time in practice, and the problem of high computational cost can be alleviated by using high-performance, or parallel computing systems [15].

5. Acknowledgement

This research was partly funded by Discovery Grant (DP0557909) from the Australian Research Council.

References

[1] H. J. Bandelt and A Dress, Reconstructing the shape of a tree from observed dissimilarity data, *Adv. Appl. Math.*, Vol. 7, 1986, pp.309-343.

[2] J. Felsenstein, Evolutionary trees from DNA sequences: A maximum likelihood approach, *J. Mol. Evol.*, 17, 1981, pp. 368-376.

[3] W. M. Fitch, A non-sequential method for constructing trees and hierarchical classifications, *J. Mol. Evol.* 18, 1981, pp. 30-37.

[4] S. Guindon and O. Gascuel, A simple, fast and accurate algorithm to estimate large phylogenies by maximum likelihood, Syst. Biol., 52, 2003, pp. 696-704.

[5] D. M. Hillis, C. Moritz and B. L. Mable, *Molecular Systematics*, 2nd ed. Sinauer Associates, Inc., 1996.

[6] K. Nieselt-Struwe and A. von Haeseler, Quartet-mapping, a generalization of the likelihood-mapping procedure, *Mol. Biol. Evol.*, 18(7), 2001, pp.1204-1219.

[7] V. Ranwez and O. Gascuel, Quartet-based phylogenetic inference: Improvements and limits, *Mol. Biol. Evol.*, 18(6), 2001, pp.1103-1116.

[8] D. R. Robinson and L. R. Foulds, An optimal way to compare additive trees using circular orders, *Journal of Computational Biology* **7**, 1981, pp. 731-744.

[9] F. J. Rohlf, Numbering binary trees with labelled terminal vertices, *Bul. Math. Biol.*, 45, 1983, pp. 33-40.

[10] A. Stamatakis, T. Ludwig, and H. Meier, RAxML-III: A fast program for maximum likelihood-based inference of large phylogenetic trees, *Bioinformatics* 21, 2005, pp. 456-463.

[11] M. Steel, The maximum likelihood point for phylogenetic tree is not unique, *Syst. Biol.*, Vol. 43, 1994, pp.560-564.

[12] K. Strimmer and A. von Haeseler, Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies, *Mol. Biol. Evol.*, 13(7), 1996, pp.964-969.

[13] L. S. Vinh and A. von Haeseler, IQPNNI: moving fast through tree space and stopping in time, *Mol. Biol. Evol.*, Vol. 21, no. 8, 2004, pp. 1565-1571.

[14] B. B. Zhou, M. Tarawneh, C. Wang, D. Chu, A. Y. Zomaya and R. P. Brent, A novel quartet-based method for phylogenetic inference, *Proceedings of IEEE International Symposium on BIBE*, Minneapolis, Oct. 2005.

[15] B. B. Zhou, D. Chu, M. Tarawneh, P. Wang, C. Wang, A. Y. Zomaya and R. P. Brent, *Parallel implementation of a quartet-based algorithm for phylogenetic analysis*, Proceedings of the Fifth IEEE International Workshop on High Performance Computational Biology, Rhodes Island, Greece, April 2006.