Neighbor Joining Approaches for Reconstructing Tandem Duplication History

Lusheng Wang
Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong, China
E-mail: lwang@cs.cityu.edu.hk

Ying Xu
Department of Computer Science
Peking University
Beijing, 100871, China
E-mail: xuying@cs.cityu.edu.hk

Louxin Zhang
Department of Mathematics and I2R
National University of Singapore
Singapore
E-mail: matzlx@nus.edu.sg

Abstract

Motivation: Genomes are replete with short sequences repeated consecutively called tandem repeats. Reconstructing duplication histories for tandem repeats may yield valuable insights into their functions and the biological mechanisms of tandem repeat creation and extension.

Results: we design and implement a set of heuristic algorithms for reconstructing tandem duplication history with neighbor-joining approaches. We prove that the algorithms are able to infer the correct duplication history under some condition. Extensive simulations show that some of our algorithms work very well. We also analyze some real data sets using our algorithms.

Availability: NJtandem is available at http://www.cs.cityu.edu.hk/$\tilde{\text{l}}$wang/tandem.

1 Introduction

Genomes, especially eukaryotic genomes, are replete with DNA repetitive regions. *Tandem repeats*, which consist of two or more contiguous copies of a particular pattern of nucleotides, are a major group of repeats, and are estimated to cover up to 10% of the human genome. Tandem repeats vary greatly, over several orders of magnitude, both in terms of the length of the repeating pattern and the number of contiguous copies. Although the functions of these repetitive regions have not been well understood, tandem repeats are receiving much attention in biological studies. Some repeats are known to play a biological role in both healthy and pathological conditions. Certain tandem repeats have been associated with self-assembly and aggregation processes (Gazit, 2002). The polymorphism of the repeats is widely used in linkage studies and DNA fingerprinting. A number of tools for search tandem repeats have been developed, e.g., Tandem Repeat Finder (Benson, 1999), Sputnik (Abajian, 1994) and TROLL (Castelo *et al.*, 2002).

Tandem repeats are believed to have arisen from tandem duplications, in which a stretch of DNA (which may contain several repeating patterns) is transformed into two adjacent copies. Since the copies may undergo additional mutations later on, approximate repeats are usually presented in a tandem array. It is interesting to know how a tandem repeat is evolved from a single copy through duplications. Because tandem duplication process is an important mechanism for generating gene family clusters on chromosomes, reconstructing tandem duplication histories allows scientists to have a better understanding of the evolution of gene families and may yield valuable insights into their functions. The duplication history of real genomic data would also be helpful to investigate the biological mechanisms of tandem repeat creation and extension.

The evolutionary study of tandem repeats started two decades ago. Fitch (1977) first formulated the duplication history of tandem repeats as a phylogeny constrained by crossovers. Recently, reconstructing tandem duplication history has been attracting increasing attention. Benson and Dong (2001) studied several versions of this problem and developed heuristic algorithms. Tang *et al.* (2001) reproposed the duplication model similar to that of Fitch. They also gave a distance-based heuristic, which was shown by simulations to work well under certain conditions. A PTAS (polynomial time approximation scheme) for the case where the duplication stretch contains only one repeating pattern was given. Jaitly *et al.* (2001) attacked the problem from a theoretic view. The problem was proved to be NP-hard. A PTAS for the same special case was independently given. Elemento *et al.* (2002) presented another method, which searches for the best duplication tree according to the maximum parsimony criterion. Zhang *et al.* (2002) gave a linear time algorithm for determining whether a given phylogeny is associated with a duplication model. They also designed an efficient heuristic for reconstructing the duplication history.

In this paper, we present a set of heuristic algorithms for reconstructing tandem duplication history with neighbor-joining approaches. We prove that some of the proposed algorithms are able to infer the correct duplication history if the distance matrix is nearly additive. Extensive simulations show that some of our algorithms work very well. Finally we analyze the duplication history of real data sets, mucin genes and ZNF genes, using our algorithms.

2 Duplication model

The duplication model for tandem repeats was proposed by Fitch (1977) and Tang *et al.* (2001) independently. Consider n repeating patterns consecutively locating on a chromosome, which are evolved from a single copy through a series of tandem duplications. A duplication replaces a stretch of DNA (which may contain several repeating patterns) with two identical and adjacent copies. If the stretch contains k repeating patterns, it is called a *k-duplication* event.

A *duplication model* M is a directed graph that has *nodes, edges*, and *blocks*. A node represents a repeat, and a directed edge between two nodes represents parent-children relationship. Node s is an *ancestor* of node t if there is a directed path from s to t. A node that has no outgoing edges is a *leaf*, and is labelled by a given sequence. A non-leaf node is called an *internal node*, which has exactly two outgoing edges. There is one special internal node designated as *root* that has only outgoing edges. Every node has one incoming edge except the root.

A block represents a duplication event. Each internal node appears in exactly one block. No node in a block is the ancestor of another node in the same block. A block corresponding to a k-duplication contains k internal nodes $v_1, v_2, \ldots, v_k$ in left-to-right order. $lc(v_i)$ and $rc(v_i)$ denote the left and right child of v_i, respectively. Then $lc(v_1), lc(v_2), \ldots, lc(v_k), rc(v_1), rc(v_2), \ldots, rc(v_k)$ are placed in the left-to-right order in the duplication model. No other edges in M cross each other. The left-to-right order of leaves should be consistent with the order of their positions on the chromosome.

A duplication model M can be viewed as a rooted binary tree T_M and some associated duplication blocks. The problem of reconstructing tandem duplication history is, to infer a duplication model for a given sequence $r_1 r_2 \ldots r_n$ of n repeating patterns, where the length of each pattern is the same.

3 Algorithms

The classical neighbor-joining Algorithms

The neighbor-joining methods have been widely used in phylogeny inference. They are agglomerative clustering algorithms, which produce a tree in a bottom-up fashion by iteratively combining taxa.

Given an $n \times n$ distance matrix for n taxa, the algorithm selects two taxa i, j that optimize some neighbor selection criterion and merges them into one new taxon u. The distance between

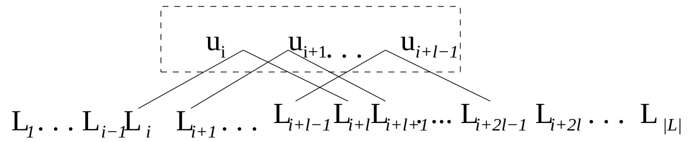


Figure 1: **Identifying and merging adjacent repeats in neighbor-joining algorithm.** l new repeats $u_i, \ldots, u_{i+l-1}$ form a duplication block. The repeat u_k $(i \le k \le i+l-1)$ is duplicated into two repeats L_k and L_{k+l}.

the new taxon u and a remaining taxon k is updated as follows:

$$D'[u,k] = \lambda D[i,k] + (1-\lambda)D[j,k].$$

The distance between any two remaining taxa is not changed. The process is repeated until only 3 taxa are left, and a phylogeny is thus constructed.

There are two popular neighbor selection criteria (Sattah and Tversky, 1977; Saitou and Nei, 1987). Sattah and Tversky's method looks for a pair of taxa (i,j) maximizing

$$C[i,j] = |\{(k,l) : D[i,j] + D[k,l] < min(D[i,k] + D[j,l], D[i,l] + D[j,k])\}|$$

where i, j, k, and l are distinct. Saitou and Nei's method looks for a pair of taxa (i,j) minimizing

$$S[i,j] = (n-2)D[i,j] - \sum_k D[i,k] - \sum_k D[j,k].$$

Neighbor-joining algorithms for duplication models

We apply neighbor-joining approach to reconstructing tandem duplication history. Given a sequence of n repeating patterns $L_1L_2...L_n$, we store the n patterns in a list L. The algorithm attempts to find a (consecutive) subsequence $L_i, \ldots, L_{i+l-1}, L_{i+l}, \ldots, L_{i+2l-1}$ that are generated in one duplication event, and merge the $2l$ repeats into l new repeats. The l new repeats form a duplication block, see figure 1. The distance between a new node u_k and others is updated as follows:

(1) $D[u_k, L_j] = \lambda D[L_k, L_j] + (1-\lambda)D[L_{k+l}, L_j]$,

(2) $D[u_k, u_j] = \lambda D[L_k, L_j] + (1-\lambda)D[L_{k+l}, L_{j+l}]$.

The algorithm repeats the process for a new list $L_1, \ldots, L_{i-1}, u_i, \ldots, u_{i+l-1}, L_{i+2l}, \ldots, L_n$ until a tree representing the duplication history is constructed. The framework of neighbor-joining algorithm is as follows.

Algorithm

Input: an $n \times n$ distance matrix D for n repeats $L = \{L_1, L_2, \ldots, L_n\}$.

Output: an unrooted tree and associated duplication blocks.

```
1. while $|L| \geq 3$
2.     for $l = 1$ to $\frac{|L|}{2}$, $i = 1$ to $|L| - 2l + 1$ do
3.         select $(i, l)$ which optimizes neighbor-selecting criterion,
4.         substitute $L_i, \ldots, L_{i+2l-1}$ with $l$ new repeats $u_i, \ldots, u_{i+l-1}$,
5.         record $(u_i, u_{i+1}, ...u_{i+l-1})$ as a duplication block,
6.         update the distance matrix,
7.         set $L = \{L_1, \ldots, L_{i-1}, u_i, \ldots, u_{i+l-1}, L_{i+2l}, \ldots, L_n\}$.
```

In practice, we let λ be $\frac{1}{2}$. The input distance matrix is calculated from the given sequence alignment according to the Jukes-Cantor model (Swofford *et al.*, 1996).

Like the neighbor-joining algorithms for phylogeny reconstruction, different neighbor-selecting criteria can be adopted in the above framework. In this paper, we introduce three criterions. The first is a variant of Sattah and Tversky (1977), and the other two are derived from Saitou and Nei (1987).

Neighbor-selecting criterion 1

$$C_1[i,j] = |\{(k,l) : D[i,j] + D[k,l] < min(D[i,k] + D[j,l], D[i,l] + D[j,k])\}|,$$

where i, j, k, and l are distinct. We look for (i, l) maximizing

$$F_1(i,l) = \frac{\sum_{j=i}^{i+l-1} C_1[j, j+l]}{l}.$$

Neighbor-selecting criterion 2

Let $S[i,j] = (n-2)D[i,j] - \sum_k D[i,k] - \sum_k D[j,k]$, and $p_i = \{j | S[i,j] = min_k S[i,k]\}$.

$$C_2[i,j] = \begin{cases} 1 \; if \; i \in p_j \wedge j \in p_i \\ 0 \; othewise. \end{cases}$$

We look for (i, l) maximizing

$$F_2(i,l) = \frac{\sum_{j=i}^{i+l-1} C_2[j, j+l]}{l}.$$

Neighbor-selecting criterion 3

$S[i,j]$ is defined as in criterion 2. We look for (i, l) minimizing

$$F_3(i,l) = \frac{\sum_{j=i}^{i+l-1} S[j, j+l]}{l}.$$

Time complexity of the algorithms

In each iteration, it is obvious that Steps 4-7 takes at most $O(n^2)$ time. There are at most n iterations. Thus the total time for Steps 4-7 is at most $O(n^3)$.

Now we analyze the time required for step 3 to find neighbors.

We consider Criterion 1 first. In the first iteration, computing all C_{ij}s needs to go through all quartets (i, j, k, l), which takes time $O(n^4)$. In the following iterations, only those quartets containing $L_i, \ldots, L_{i+2l-1}$ and new repeats $u_i, \ldots, u_{i+l-1}$ need to be recomputed, and the total the number is $O(ln^3)$. The sum of l in all iterations is $n - 3$, so the total time for computing C_{ij}s is $O(n^4)$. Given C_{ij}s, we need to further compute $F_1(i, l)$s. There are $O(n^2)$ possible combinations of (i, l) in total. Since $F_1(i + 1, l)$ can be computed from $F_1(i, l)$ in constant time by subtracting $C_1[i, i + l]/l$ and adding $C_1[i + l, i + 2l]/l$, computing all $F_1(i, l)$s in one iteration takes $O(n^2)$ time, and the total time for computing $F_1(i, l)$s is at most $O(n^3)$. Therefore, the total time of the algorithm under Criterion 1 is $O(n^4)$.

For Criterion 2, computing S_{ij}, p_i, and C_{ij} in each iteration takes time $O(n^2)$. Given C_{ij}'s, to compute all $F_2(i, l)$'s in one iteration takes $O(n^2)$ time (the argument is the same with that of Criterion 1). Therefore, the time complexity of the whole algorithm under Criterion 2 is $O(n^3)$.

Similarly, the time complexity under Criterion 3 is $O(n^3)$.

4 Performance when the given matrix is nearly additive

Given a tree T containing n leaves, we have an $n \times n$ distance matrix D^T of T, where $D^T[i, j]$ is the total length of the path from leaf i to leaf j in T. Let e be an edge of T and $l(e)$ the length of e. A matrix D is *nearly additive* if there exists a binary tree T satisfying

$$|D - D^T| < min_{e \in T} \frac{l(e)}{2},$$

where $|D - D^T|$, the *error* between two matrices, is defined as $max_{i,j}|D[i, j] - D^T[i, j]|$. D is said to be *nearly additive* with respect to T. Atteson (1999) and Wang and Gusfield (1998) independently proved that

Theorem 1 *The binary tree corresponding to a nearly additive matrix is unique.*

It is known that when the given distance matrix is nearly additive, both versions of the neighbor joining algorithm given in (Sattah and Tversky, 1977) and (Saitou and Nei, 1987) are able to output this unique tree (Atteson, 1999).

Given a duplication model M, we also have a binary tree T_M if the blocks are ignored. An $n \times n$ distance matrix D is *nearly additive* with respect to a duplication model M if D is nearly additive with respect to T_M. In this section, we prove that our neighbor-joining algorithms for tandem duplication can also output the unique tree if the matrix is nearly additive with respect to a duplication model. More precisely, we have

Theorem 2 *Given a matrix nearly additive with respect to a duplication model M, the neighbor-joining algorithm adopting neighbor-selecting criterion 1 or 2 is able to output T_M.*

5 Results

We implemented the above algorithms using C++. It takes a multiple sequence alignment as input, computes a distance matrix from the sequences according to the Jukes-Cantor model (Swofford *et al.*, 1996), and infers duplication models using neighbor-joining methods. Three algorithms were implemented using criterions 1, 2, and 3. Users can choose one or all of three algorithms to generate the duplication model.

Extensive simulations have been conducted to evaluate the performance of these algorithms. We also apply our method to real data sets: tandem repeats in the human mucin gene MUC5B and zinc-finger (ZNF) genes. These sequences are strongly believed to be generated by tandem duplications.

5.1 Simulation results

Simulations were conducted to evaluate the performance of the neighbor-joining algorithms.

A simulation procedure works as follows. First, a random tree is generated using DNAtree, a computer program that simulates the branching of an evolutionary tree, using a model of random branching of lineages (Kuhner and Felsenstein, 1994). Then a duplication model M is obtained from the tree by randomly placing multiple duplication blocks according to some pre-defined frequency, and branch lengths are assigned on M so that the model satisfies the molecular clock assumption. Given the duplication model, a set of (aligned) sequences, one sequence for each leaf, are then produced according to the generated duplication model by Seq-Gen (Rambaut and Grassly, 1997). Our program takes the sequence alignment as input, and generates duplication models inferred by three algorithms. Finally, these models output by our program are compared with the original model M to evaluate the algorithms.

We measure the performance by the correct rate of duplication blocks, i.e. the number of duplication blocks correctly recovered by the algorithm divided by the total number of duplication blocks in the original duplication model M. Generally, we say a duplication block $A = (a_1, ...a_n)$ in M is correctly recovered in another duplication model M', if there exists a duplication block $B = (b_1, ...b_n)$ in M', such that for any i, $1 \leq i \leq n$, the subtree rooted at $l(a_i)$, the left children of a_i, has the same set of leaves as the subtree rooted at $l(b_i)$, and so do their right subtrees.

However, we also consider the degenerate cases in evaluation. Sometimes, different binary trees can represent the same evolutionary history when the data are degenerated. For example, the two binary trees in Figure 2 (a) and (b) represent the same fact that the string AA changes to the two strings AC and CA. Both cases degenerate into (c). In this case, it is not reasonable to consider tree (b) as incorrect if (a) is given as the original model. Therefore, when comparing a resulting duplication model M' with the original model M, if two subtrees from M and M' have different topologies with the same tree scores (see Gusfield (1997) for the definition of tree score), then we

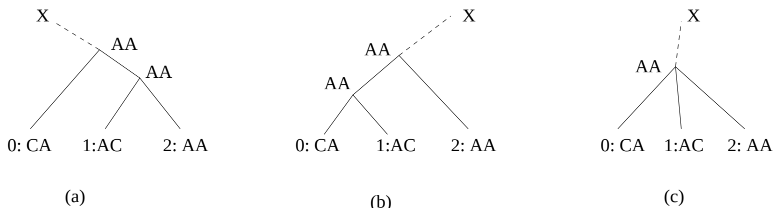


Figure 2: **Three trees represent the same evolutionary history.** The string AA is assigned to the internal nodes. X represents the rest part of the tree.

consider the two subtrees to be equally good, and we do not count the duplication blocks in the subtrees as wrong even if they are not identical.

We ran our program against various data sets generated by the above procedure. Three parameters need to be specified in the procedure: (1) repeating pattern length, (2) copy number: the number of repeating patterns in a tandem repeat, and (3) mutation rate: a parameter indicating the frequency of mutation events, used by Seq-Gen when generating repeat sequences. We conducted simulations using different parameter settings: mutation rate is from 0.05 to 0.50 (increments of 0.05), copy number from 10 to 60 (increments of 5), and repeating pattern length from 100 to 800 (increments of 100). For each combination of parameter values, 50 data sets were generated, and the performance is measured by the mean correct rate of the 50 runs. Figure 3 compares the performance of different neighbor-joining algorithms under different parameters. (Algorithm 1 is the neighbor-joining algorithm using Criterion 1, Algorithm 2 uses Criterion 2, and Algorithm 3 uses Criterion 3.)

It is interesting to notice that though no theoretical results could be derived for Algorithm 3, Algorithm 3 almost always beats the other two algorithms in practice. Algorithm 3 consistently performs a high correct rate (mostly more than 90%) and is slightly better than Algorithm 2, while Algorithm 1 shows relatively low performance (below 30%). The two cases that Algorithm 2 outperforms Algorithm 3 is when copy number is small (Figure 3 (a)) and when repeating pattern length is large (Figure 3 (c)).

Figure 3 (a)-(c) also show the effects of copy number, mutation rate, and repeating pattern length on the performance of the algorithms, with other parameters fixed. We observe that the performance of the three algorithm all increases as repeating pattern length increases (Figure 3 (a)), and copy number decreases (Figure 3 (c)). (The effect of repeating pattern length on performance of Algorithm 1 is small.) On the other hand, mutation rate shows little impact on performance (Figure 3 (b)).

Figure 3 (d) shows the performance of Algorithm 3 as copy number and duplication block size changes. Three curves in the figure correspond to the following three duplication block sizes: (1) all duplication blocks are single-duplication; (2) around half internal nodes are in multiple duplication blocks; (3)place multiple duplication blocks whenever possible when simulating evolutionary history. (Figure (a) to (c) are all obtained under condition (2).) We see that the algorithm performs better when duplication size is small, but the difference is not large. The effects of duplication size on performance of Algorithms 1 and 2 have similar patterns.

5.2 Mucin gene MUC5B

Mucus consists mainly of mucin proteins. Four mucin genes, MUC6, MUC2, MUC5AC and MUC5B, have been identified, each containing a central part of tandem repeats. In the central part of MUC5B is a single large exon of about 11kbp containing four repeats (Desseyn *et al.*, 1997). Each repeat consists of an R-subdomain (11 or 17 tandem repeats of a motif of 29 amino acids), an R-End subdomain, and a Cys-subdomain. Another R-subdomain of 23 tandem repeats of 29 amino acids follows the fourth repeat. This suggests that the central part of MUC5B arose from tandem duplications.

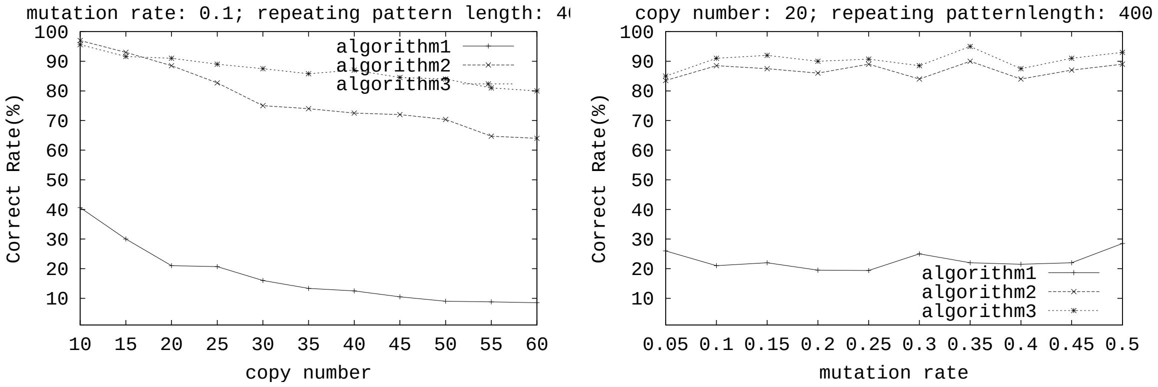
We applied our algorithms on the third repeat segment RIII, which consists of 17 tandem repeats of an irregular motif of 29 amino acids rich in Ser and Thr. The amino acid sequences of MUC5B were taken from Desseyn *et al.* (2000) and aligned using an alignment program at Michigan Tech http://genome.cs.mtu.edu/map/map.html (Huang, 1994). The duplication models output by the three algorithms are illustrated in Figure 4 (a), (b) and (c), respectively.

We can see that the duplication models inferred by Algorithms 2 and 3 are relatively close (have 4 duplication blocks in common, including one block of size 5), while model 1 is more diverse. The tree produced by Algorithm 1 has a larger cost (162) than those produced by Algorithm 2 (138) and Algorithm 3 (142), and is thus less convincing according to the parsimony principle. These facts, combined with the conclusion drawn from the simulation results, indicate that the duplication histories inferred by Algorithms 2 and 3 are more reliable. The 5-duplication block, which contains repeats 1-11, is not only confirmed by two of our algorithms, but also appears in the duplication model inferred in Zhang *et al.* (2002) using a different approach. Therefore, we believe this local duplication history is reliable.

5.3 ZNF gene families

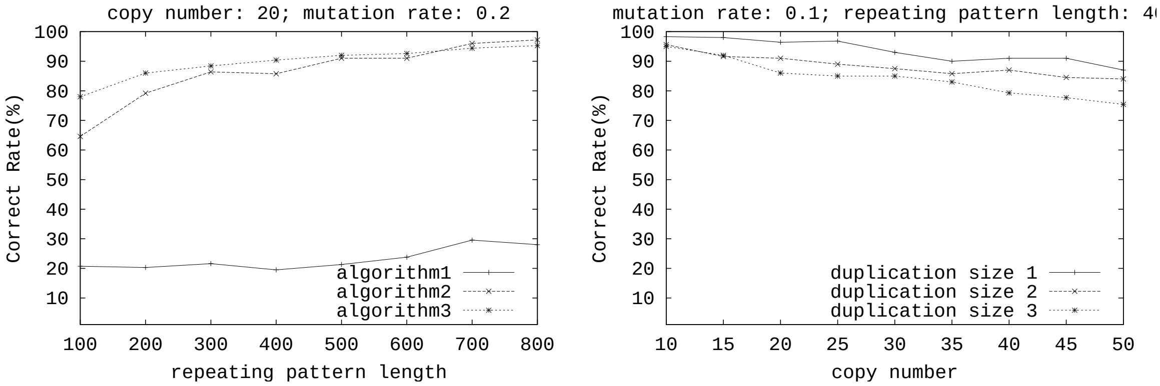
Human genome contains roughly 700 Kruppel-type (C2H2) zinc-finger (ZNF) genes, which encode putative transcription factors. We choose the gene cluster ZNF45 and obtained its gene-duplication history using our algorithms. The members of the ZNF45 family have different numbers of zinc fingers. *In situ* tandem duplication events are likely to have given rise to this features.

The amino acid sequences of 16 genes in the gene family ZNF45 were aligned using the alignment program of Huang (1994), and the result alignment sequences were given as input to neighbor-joining algorithms. The duplication models output by the three algorithms are illustrated in Figure 5 (a), (b) and (c), respectively. The three duplication models, as well as the models inferred



(a) Performance under different copy numbers

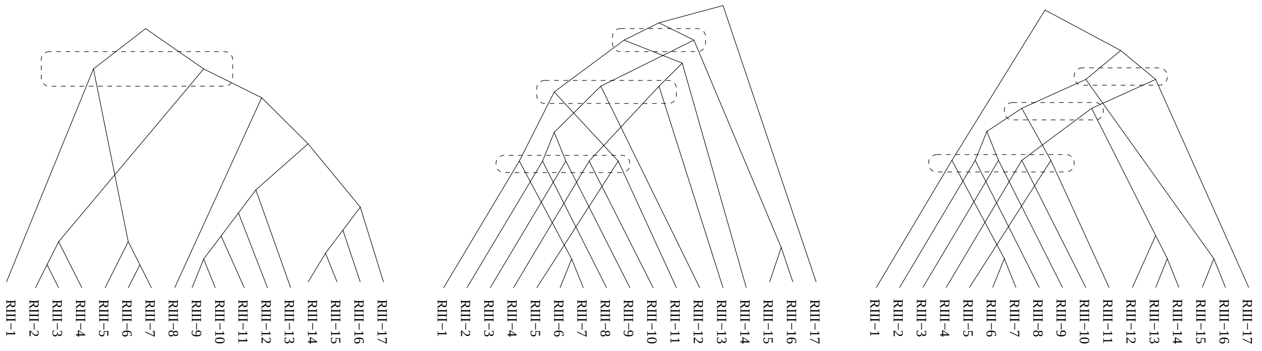
(b) Performance under different mutation rates



(c) Performance under different repeating pattern lengths

(d) Performance under different duplication sizes

Figure 3: **Comparison of the performance of three neighbor-joining algorithms.** Figure (a) shows the performance of the three algorithms under different copy numbers, with the other parameters fixed; (b) shows the performance under different mutation rates; (c) shows the performance under different repeating pattern lengths. Figure (d) shows the performance of algorithm3 as copy number and duplication block size changes.



(a) Duplication model1

(b) Duplication model2

(c) Duplication model3

Figure 4: **Duplication histories of RIII in MUC5B central part inferred by three neighbor-joining algorithms.** (a), (b) and (c) are the duplication models inferred by Algorithm 1, 2 and 3, respectively. The order of leaves from left to right is the same as the order of their positions on chromosome. Multiple duplications are represented by boxes.

in Tang *et al.* (2001) and Zhang *et al.* (2002) all agree on some local histories, for example, the neighbor relationship of (ZNF221, ZNF155), (ZNF230, ZNF222, ZNF223), (ZNF224, ZNF225) and (ZNF234, ZNF226). The five duplication models also suggest that there is a 2-duplication in early history, although they do not quite agree with how these copies evolve afterwards.

References

[1] Abajian,C. (1994) Sputnik. http://abajian.net/sputnik/.

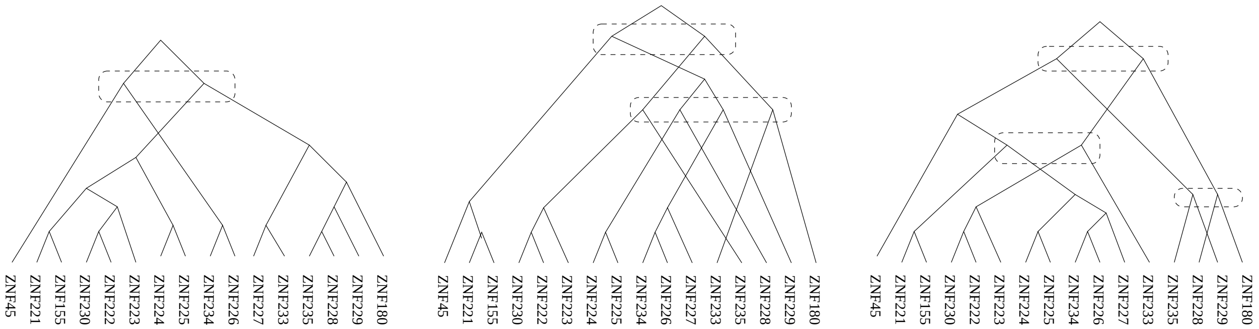
[2] Atteson,K. (1999) The performance of neighbor-joining methods of phylogenetic reconstruction, *Algorithmica*, 25, 251-278.

[3] Benson,G. (1999) Tandem repeats finder: a program to analyze DNA sequences, *Nucleic Acids Res.*, 27, 573-580.

[4] Benson,G. and Dong,L. (1999) Reconstructing the duplication history of a tandem repeat, *Proc. of the 7th International Conference on Intelligent Systems for Molecular Biology* (ISMB99), 44-53.

[5] Castelo,A., Martins,W. and Gao,G. (2002) TROLL: Tandem Repeat Occurrence Locator, *Bioinformatics*, 18: 634-636.

[6] Desseyn,J.L., Guyonnet-Duperat,V., Porchet,N., Aubert,J.P. and Laine,A. (1997) Human mucin gene MUC5B, the 10.7 -kb large central exon encodes various alternate subdomains



(a) Duplication model1 (b) Duplication model2 (c) Duplication model3

Figure 5: **Duplication histories of ZNF45 inferred by three neighbor-joining algorithms.** (a), (b) and (c) are the duplication models inferred by Algorithms 1, 2 and 3, respectively. The order of the leaves from left to right is the same as the order of their positions on chromosome. Multiple duplications are represented by boxes.

resulting in a super-repeat. Structural evidence for a 11p15.5 gene family. *J. Biol. Chem.*, 272, 3168-3178.

[7] Desseyn,J.L., Aubert,J.P., Porchet,N. and Laine,A. (2000) Evolution of the large secreted gel-forming mucins, *Mol. Biol. Evol.*, 17, 1175-1184.

[8] Elemento,O., Gascuel,O. and Lefranc,M. (2002) Reconstructing the duplication history of tandemly repeated genes, *Mol. Biol. Evol.*, 19, 278-288.

[9] Fitch,W.(1977) Phylogenies constrained by cross-over process as illustrated by human hemoglobins in a thirteen cycle, eleven amoni-acid repeat in human apolipoprotein A-I, *Genetics*, 86, 623-644.

[10] Gazit,E. (2002) Global analysis of tandem aromatic octapeptide repeats: The significance of the aromaticCglycine motif, *Bioinformatics*, 18, 880-883.

[11] Gusfield, D. (1997) Algorithms on strings, trees, and sequences, Cambridge University Press.

[12] Huang,X. (1994) On Global Sequence Alignment, *Comput. Appl. in Biosci*, 10, 227-235.

[13] Jaitly,D., Kearney,P., Lin,G. and Ma,B. (2001) Methods for reconstructing the history of tandem repeats and their application to the human genome, *Journal of Computer and System Sciences*, to appear.

[14] Kuhner,M.K. and Felsenstein,J. (1994) A simulation comparison of phylogeny alogirthms under equal and unequal evolutionary rates, *Mol. Biol. Evol.*, 11:459-468.

[15] Rambaut,A. and Grassly,N.C. (1997) Seq-Gen: An application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees, *Comput. Appl. Biosci.*, 13, 235-238.

[16] Saitou,N. and Nei,M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees, *Mol. Biol. Evol.*, 4: 406-425.

[17] Sattah,S. and Tversky,A. (1977) Additive similarity tree, *Psychometrika*, 42:319-345.

[18] Swofford,D., Olsen,G., Waddell,P. and Hillis,D. (1996) Phylogeny inference, chapter 11 in *Molecular systematics*, Sinauer Associates, Inc. Publishers.

[19] Tang,M., Waterman,M. and Yooseph,S. (2001) Zinc finger gene clusters and tandem gene duplication, *RECOMB01*, 297-304

[20] Wang,L. and Gusfield,D. (1998) Constructing Additive Trees When the Error Is Small, *Journal of Computational Biology*, 5, 137-134.

[21] Zhang,L., Ma,B., Wang,L. and Xu,Y. (2002) Gready method for inferring tandem duplication history, *Bioinformatics*, to appear.

Appendix: The proof of Theorem 2

Consider Criterion 1 first. Atteson (1997) proved the following lemma and corollary.

Lemma 3 *Given a matrix D nearly additive with respect to a tree T, and four distinct leaves i, j, k, l, if there exists an edge e in T such that L_i and L_j are contained in the same component of $T - e$, and L_k and L_l are contained in the other component, then* $D[i,j] + D[k,l] < min(D[i,k] + D[j,l], D[i,l] + D[j,k])$.

Corollary 4 *Given a matrix D nearly additive with respect to a tree T, if i and j are neighbors in T, then $C_1[i,j] = (n-2)(n-3)/2$; otherwise, $C_1[i,j] < (n-2)(n-3)/2$.*

Lemma 5 *For a matrix D $(n > 3)$ nearly additive with respect to a duplication model M, let (i,l) optimize Criterion 1. Then (L_i, L_{i+l}), …, (L_{i+l-1}, L_{i+2l-1}) are neighbors in T_M.*

Proof. Consider a duplication block $(v_1, \ldots, v_l)$ at the bottom of T_M whose children are all leaves. Then $C_1[lc(v_i), rc(v_i)] = (n-2)(n-3)/2$ according to Corollary 4. It is easy to verify that $F_1(lc(v_1), l) = (n-2)(n-3)/2$. Thus, we have

$$max_{i,l} F_1(i,l) \geq (n-2)(n-3)/2.$$

For any (i,l) satisfying $F_1(i,l) \geq (n-2)(n-3)/2$, we have

$$C_1[j, j+l] = (n-2)(n-3)/2$$

for $i \leq j \leq i+l-1$, since $max_{i,j} C_1[i,j] = (n-2)(n-3)/2$. According to Corollary 4, L_j and L_{j+l} are neighbors.

Therefore, for any (i,l) maximizing $F_1(i,l)$, $(L_i, L_{i+l}), \ldots, (L_{i+l-1}, L_{i+2l-1})$ are neighbors.$\Box$

Now, we show that the update of matrix according to the algorithm is correct.

Let D be an $n \times n$ $(n > 3)$ matrix that is nearly additive with respect to a duplication model M, and (i,l) optimize neighbor-selecting criterion 1 or criterion 2 (see Lemma 7 in this case) and thus $(L_i, L_{i+l}), \ldots, (L_{i+l-1}, L_{i+2l-1})$ are neighbors in T_M. Let D' be a $(n-l) \times (n-l)$ matrix obtained from the algorithm after substituting $L_i, \ldots, L_{i+2l-1}$ with l new repeats $u_i, \ldots, u_{i+l-1}$ in the algorithm. Let T' be the tree obtained from T_M as follows: (1) delete the $2l$ leaves $L_i, \ldots, L_{i+2l-1}$ in T and thus the parents $u_i, \ldots, u_{i+l-1}$ of $L_i, \ldots, L_{i+2l-1}$ become leaves in T' (2) let the parent of u_j be x_j in T' for $j = i, i+1, ..., i+l$, the length of edge (x_j, u_j) in T' is updated as $\lambda l(x_j, L_x) + (1-\lambda) l(x_j, L_y)$, where $l(x_j, L_x)$ and $l(x_j, L_y)$ are the lengths of paths from x_j to L_x and from x_j to L_y in T, respectively. (3) the lengths of other edges in T' remain the same as those in T.

Lemma 6 *D' is nearly additive with respect to T'.*

Proof. Consider the distance matrix $D^{T'}$ for T'. By definition of T', $D^{T'}$ is updated from D^T as follows:

$$\begin{aligned}
D^{T'}[u_k, L_j] &= D^T[u_k, L_j] + \lambda l(u_k, L_k) + (1-\lambda) l(u_k, L_{k+l}) \\
&= \lambda(D^T[u_k, L_j] + l(u_k, L_k)) + (1-\lambda)(D^T[u_k, L_j] + l(u_k, L_{k+l})) \\
&= \lambda D^T[L_k, L_j] + (1-\lambda) D^T[L_{k+l}, L_j]. \\
D^{T'}[u_k, u_j] &= D^T[u_k, u_j] + \lambda l(u_k, L_k) + (1-\lambda) l(u_k, L_{k+l}) + \lambda l(u_j, L_j) + (1-\lambda) l(u_j, L_{j+l}) \\
&= \lambda(D^T[u_k, u_j] + l(u_k, L_k) + l(u_j, L_j)) + (1-\lambda)(D^T[u_k, u_j] + l(u_k, L_{k+l}) + l(u_j, L_{j+l})) \\
&= \lambda D^T[L_k, L_j] + (1-\lambda) D^T[L_{k+l}, L_{j+l}]
\end{aligned}$$

The distances between other leaves are not changed in T', i.e. $D^{T'}[L_k, L_j] = D^T[L_k, L_j]$.

Now we want to prove that $|D' - D^{T'}| \leq |D - D^T|$.

Since the path between two old leaves L_k and L_j is not changed, $|D'[L_k, L_j] - D^{T'}[L_k, L_j]| = |D[L_k, L_j] - D^T[L_k, L_j]|$.

When new leaves are involved,

$$\begin{aligned}
&|D'[u_k, L_j] - D^{T'}[u_k, L_j]| \\
&\quad = \quad |(\lambda D[L_k, L_j] + (1-\lambda) D[L_{k+l}, L_j]) - (\lambda D^T[L_k, L_j] + (1-\lambda) D^T[L_{k+l}, L_j])| \\
&\quad = \quad |(\lambda (D[L_k, L_j] - D^T[L_k, L_j]) + (1-\lambda)(D[L_{k+l}, L_j] - D^T[L_{k+l}, L_j])| \\
&\quad \leq \quad \lambda |D - D^T| + (1-\lambda) |D - D^T| \\
&\quad = \quad |D - D^T| \\
&|D'[u_k, u_j] - D^{T'}[u_k, u_j]| \\
&\quad = \quad |(\lambda D[L_k, L_j] + (1-\lambda) D[L_{k+l}, L_{j+l}]) - (\lambda D^T[L_k, L_j] + (1-\lambda) D^T[L_{k+l}, L_{j+l}])|
\end{aligned}$$

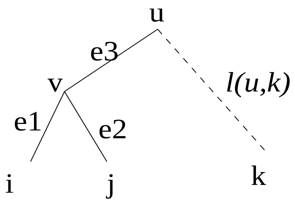


Figure 6: **Illustration of proof in Lemma 8.**

$$\begin{aligned} &= |(\lambda(D[L_k, L_j] - D^T[L_k, L_j]) + (1-\lambda)(D[L_{k+l}, L_{j+l}] - D^T[L_{k+l}, L_{j+l}])| \\ &\leq |D - D^T| \end{aligned}$$

From the construction of T', $min_{e \in T} \frac{l(e)}{2} \leq min_{e \in T'} \frac{l(e)}{2}$. Thus, we have

$$|D' - D^{T'}| \leq |D - D^T| < min_{e \in T} \frac{l(e)}{2} \leq min_{e \in T'} \frac{l(e)}{2}.$$

The second inequality holds due to the condition that input distance matrix D is nearly additive. Therefore, we can conclude that D' is nearly additive with respect to T'. □

Obviously, T' is also associated with a duplication model. From Lemma 5 and Lemma 6, we can immediately conclude that Theorem 2 holds for Criterion 1.

To prove that Theorem 2 holds for Criterion 2, we only have to prove the following lemma that is analogous to Lemma 5.

Lemma 7 *For a matrix ($n > 3$) nearly additive with respect to a duplication model M , let (i, l) optimize Criterion 2. Then (L_i, L_{i+l}), …, (L_{i+l-1}, L_{i+2l-1}) are neighbors in T_M.*

To prove Lemma 7, we need

Lemma 8 *Given a matrix D nearly additive with respect to a binary tree T, if leaves i and j are neighbors in T, then $S[i, j] < min(S[i, k], S[j, k])$ for any $k \neq i, j$.*

Proof. We only have to prove $S[i, j] < S[i, k]$. The relationship of i, j and k is illustrated in figure 6.

Let $S^T[i, j]$ denote the value of $S[i, j]$ under distance matrix D^T. By definition, we have

$$\begin{aligned} S^T[i,k] - S^T[i,j] &= ((n-2)D^T[i,k] - \sum_p D^T[i,p] - \sum_p D^T[k,p]) \\ &- ((n-2)D^T[i,j] - \sum_p D^T[i,p] - \sum_p D^T[j,p]) \\ &= (n-2)(D^T[i,k] - D^T[i,j]) - \sum_p (D^T[k,p] - D^T[j,p]) \\ &= (n-3)(D^T[i,k] - D^T[i,j]) - \sum\ _{p \neq i,j,k}(D^T[k,p] - D^T[j,p]). \qquad (1) \end{aligned}$$

From Figure 6,

$$D^T[i,k] - D^T[i,j] = (e_1 + e_3 + l(u,k)) - (e_1 + e_2) = e_3 + l(u,k) - e_2 \tag{2}$$

and for any $p \neq i, j, k$,

$$D^T[k,p] - D^T[j,p] = D^T[k,p] - (D^T[u,p] + e_2 + e_3) \leq l(u,k) - e_2 - e_3. \tag{3}$$

Substituting (2) and (3) into (1), we have

$$S^T[i,k] - S^T[i,j] \geq (n-3)(e_3 + l(u,k) - e_2) - (n-3)(l(u,k) - e_2 - e_3) = 2e_3(n-3). \tag{4}$$

Atteson(1999) proved that, in a matrix D nearly additive with respect to tree T, $(S[i,k] - S[i,j]) - (S^T[i,k] - S^T[i,j]) \geq -4(n-3)|D - D^T|$. Thus, we have

$$S[i,k] - S[i,j] \geq (S^T[i,k] - S^T[i,j]) - 4(n-3)|D - D^T|. \tag{5}$$

Substituting (4) into (5),

$$S[i,k] - S[i,j] \geq 2e_3(n-3) - 4(n-3)|D - D^T| > 0.$$

The last inequality holds because D is nearly additive with respect to T. □

From Lemma 8 and the definition of C_2, we immediately have

Corollary 9 *Given a matrix D nearly additive with respect to a tree T, if i and j are neighbors in T, then $C_2[i,j] = 1$; otherwise, $C_2[i,j] = 0$.*

The proof of Lemma 7

Proof. First, we prove that $max_{i,l} F_2(i,l) \geq 1$. Consider a duplication block $(v_1, \ldots, v_l)$ at the bottom of T_M whose children are all leaves. $C_2[lc(v_j), rc(v_j)] = 1$ according to Corollary 9. Hence,

$$F_2(lc(v_1), l) = \frac{\sum_{j=1}^{l} C_2[lc(v_j), rc(v_j)]}{l} = 1.$$

For any (i,l) satisfying $F_2(i,l) \geq 1$, we have $C_2[j, j+l] = 1$ for $i \leq j \leq i+l-1$, since $max_{i,j} C_2[i,j] = 1$. According to Corollary 9, L_j and L_{j+l} are neighbors in T_M.

Therefore, Lemma 7 holds. □

From Lemma 7 and Lemma 6, we can immediately conclude that Theorem 2 holds for Criterion 2.