

An exact and polynomial distance-based algorithm to reconstruct single copy tandem duplication trees

Olivier Elemento^{1,2} and Olivier Gascuel¹

¹ Département d'Informatique Fondamentale et Applications, LIRMM, 161 rue Ada, 34392, Montpellier, France, <http://www.lirmm.fr/w3ifa/MAAS/>, {elemento, gascuel}@lirmm.fr

² IMGT, the international ImMunoGeneTics database, <http://imgt.cines.fr>, Laboratoire d'Immunogénétique Moléculaire, LIGM, Université Montpellier II, UPR CNRS 1142, IGH, 141 rue de la Cardonille, 34396, Montpellier Cedex 5, France

Abstract The problem of reconstructing the duplication tree of a set of tandemly repeated sequences which are supposed to have arisen by unequal recombination, was first introduced by Fitch (1977), and has recently received a lot of attention. In this paper, we deal with the restricted problem of reconstructing single copy duplication trees. We describe an exact and polynomial distance based algorithm for solving this problem, which has previously been shown to be NP-hard within a parsimony framework (like almost every evolutionary reconstruction problem). This algorithm is based on the minimum evolution principle, and thus involves selecting the shortest tree as being the correct duplication tree. After presenting the underlying mathematical concepts behind the minimum evolution principle, and some of its benefits (such as statistical consistency), we provide a new recurrence equation to estimate the tree length using ordinary least-squares, given a matrix of pairwise distances between the copies. We then show how this equation naturally forms the dynamic programming framework on which our algorithm is based, and provide an implementation in $O(n^3)$ time and $O(n^2)$ space, where n is the number of copies.

1 Introduction

Tandemly repeated DNA sequences consist of two or more adjacent copies of a stretch of DNA. They arise from tandem duplication, in which a sequence of DNA (which may itself contain several copies) is transformed into two adjacent and identical sequences. Since copies are then free to evolve independently and are likely to undergo additional mutation events, they become approximate over time. Unequal recombination is widely viewed as the predominant biological mechanism responsible for the production of tandemly repeated sequences [1,2,3,4,5,6], at least when the basic repeated motif is large (*i.e.* corresponds to minisatellites or genes). The problem of reconstructing the duplication history of tandemly repeated sequences was first considered by Fitch in 1977 [3]. It has not received much attention until recently, probably due to the lack of available repeated sequence data, and also because there has been no dedicated computer program available to reconstruct duplication histories from sequence data. With the sequencing of human genome, and the ongoing sequencing of several other species, this problem has gained a lot of attention. The reason is that fast and efficient methods for reconstructing the duplication history of these tandemly repeated sequences would be important tools to study genome evolution. They should provide deeper insights into the processes, dynamics and mechanisms of gene duplications, which is considered as one of the major evolutionary forces at the genome level [1].

Most of the recent studies have been devoted to repeated sequences generated by single copy duplication events [7,8,9,10]. Indeed, the mechanism of unequal recombination allows simultaneous duplication of several copies, but there is now evidence [3,5,6,7,8] that single copy duplications are predominant over multiple copies duplications, at least with minisatellites or tandemly repeated genes. For example, one of the most famous tandemly arranged gene families, the Antennapedia (*antp*)-class homeobox genes, is assumed to have arisen through repetitive single copy duplications [11].

The series of duplications that has given rise to tandemly repeated sequences can be represented by way of a “duplication tree”, which we formally describe below. A duplication tree which only contains single copy duplications is simply called a “single copy duplication tree”. Reconstructing optimal single copy duplication trees has been shown to be NP-hard within a parsimony framework [10], and several authors described approximation algorithms. Benson and Dong [7] developed a greedy algorithm for reconstructing single copy duplication trees, based on the parsimony criterion. Using simulations, they showed that their algorithm performs better than approximation algorithms based on minimum ordered spanning trees, which themselves guarantee a performance ratio of 2. More recently, Tang et al. [8] described a dynamic programming algorithm within a parsimony framework, based on the lifting technique [12], for the same problem. Then, they showed that the cost of the obtained tree is at most twice higher than the cost of the most parsimonious tree. Later, Tang et al. [9] and Jaitly et al. [10] independently described polynomial time approximation schemes (PTAS) for the single copy problem (within the same parsimony framework), also obtained using the lifting technique combined with local optimization and a dynamic programming framework.

In this paper, we present an exact and polynomial $O(n^3)$ time and $O(n^2)$ space algorithm for reconstructing the optimal single copy duplication tree, where n is the number of copies. Our algorithm is based on the minimum evolution principle [13,14], and uses as input a matrix of pairwise evolutionary distances, calculated from a set of ordered nucleotide or protein sequences. The minimum evolution principle involves selecting the tree with shortest least-squares length estimate as being the correct tree. Due to the use of this principle, our reconstruction algorithm is consistent [14,15], as opposed to parsimony methods which were shown inconsistent by Felsenstein [16]. The content of this paper is organized as follows. First we describe the duplication model, *i.e.* the characteristics of the mathematical objects we aim at reconstructing. Then we describe the minimum evolution framework on which our algorithm is based and provide a novel recurrence equation for estimating the length of any given tree, from a distance matrix between copies. Using this equation, we describe a dynamic programming algorithm to solve the single copy duplication tree problem under the minimum evolution principle.

2 Duplication model

Assuming unequal recombination as the sole mechanism responsible in generating the copies, Fitch [3], and more recently Tang et al. [8,9] and Elemento et al. [5,6] independently introduced the following duplication model. A duplication history (Figure 1(a) and 1(b)) is a rooted tree with n labelled and ordered leaves denoted as $(1,2,3,\dots,n)$, in which internal nodes correspond to duplication events. In a real duplication history, the time intervals between consecutive duplications are completely known, and the internal nodes are ordered from top to bottom according to the moment they occurred in the course of evolution. However, in the absence of molecular clock (which is almost always the case), both the order between the duplication events of two different lineages and the root location are impossible to recover from the sequences. In this case, we are only able to infer a *duplication tree* (Figure 1(c)), *i.e.* an unrooted tree with ordered leaves, whose topology is compatible with at least one duplication history. Recovering the position of the root can sometimes be achieved, through the use of rooting procedures (outgroups, midpoint, etc.), and creates a *rooted duplication tree* (Figure 1(d)).

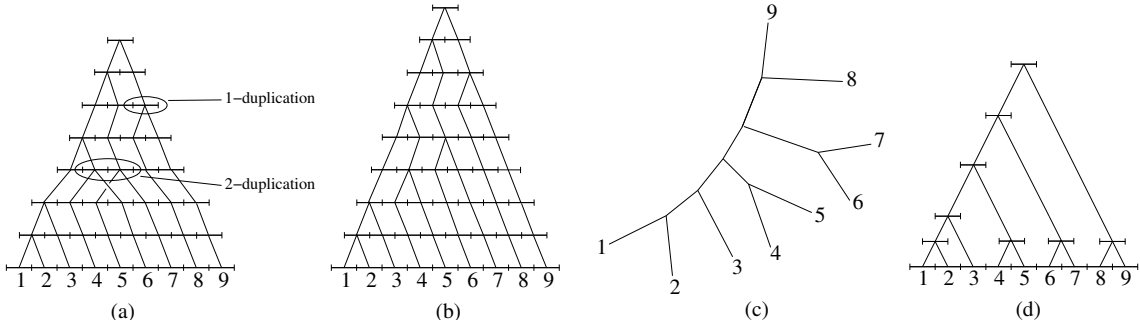


Figure 1. (a) duplication history, (b) single copy duplication history, (c) single copy duplication tree and (d) single copy rooted duplication tree

A duplicated fragment may only contain a single copy, in which case we say that the duplication event is a 1-duplication (or a single copy duplication). It may also contain 2, 3 or k copies, in which case we call the duplication event a 2-, 3- or k -duplication. When a rooted duplication tree only contains 1-duplication events (such as Figure 1(d)), it is analogous to a binary search tree. Consequently, the number of distinct single copy rooted duplication trees is equal to the number of binary search trees, which is given by the Catalan recursion [17] :

$$C_n = \sum_{k=1}^{n-1} C_k C_{n-k} = \frac{(2n)!}{n!(n+1)!} \sim \frac{4^n}{\sqrt{\pi n^{3/2}}}.$$

It was shown in [3] and later in [6] that the general duplication model constraints the root of a duplication tree to be located somewhere in the tree along the path between the most distant copies on the locus, but (generally) not anywhere due to the presence of multiple duplications. It is easy to see that a single copy duplication tree can be rooted anywhere along the path between the two most distant copies. Suppose that we systematically root duplication trees on the rightmost branch, *i.e.* the branch associated with label n . In this situation, the left subtree is a single copy rooted duplication tree with $n-1$ leaves. Therefore, the number of distinct single copy unrooted duplication trees with n leaves is simply equal to C_{n-1} . Since this number is exponential (see above), exploring the landscape of single copy duplication trees in search for the optimal one cannot be done using a trivial algorithm.

A single copy rooted duplication tree $X_{1,n}$, whose leaves are labelled with the ordered set of copies $(1, 2, 3, \dots, n)$, is obtained by combining two rooted subtrees $X_{1,p}$ and $X_{p+1,n}$ whose leaves are labelled with the ordered sets $(1, 2, 3, \dots, p)$ and $(p+1, p+2, \dots, n)$, respectively. Identically, a single copy unrooted duplication tree $X_{1,n}$ is obtained by combining three rooted subtrees $X_{1,p}$, $X_{p+1,q}$ and $X_{q+1,n}$ with $1 \leq p < q < n$. In the rest of this paper, the ordered set $(p, p+1, \dots, q)$ is denoted as $[p, q]$, while, depending on the context, $X_{p,q}$ refers to a rooted tree on $[p, q]$ or to $[p, q]$ itself.

3 Minimum evolution principle and least-squares tree length estimation

3.1 The minimum evolution principle

The minimum evolution (ME) principle [13,14] involves selecting the shortest tree as being the correct phylogeny. In most cases, the tree length is equal to the sum of branch lengths and these branch lengths are estimated by minimizing a least-squares criterion. Inferring optimal phylogenies under the ME principle is different from directly inferring optimal phylogenies using least-squares, which has previously been shown to be NP-hard [18]. The problem of inferring optimal phylogenies

within the ME principle is also suspected to be NP-hard, but not classified, to the best of our knowledge. Nonetheless, the ME principle forms the basis of several phylogenetic reconstruction methods, generally based on greedy heuristics. Among them is the Neighbor-Joining (NJ) algorithm [19] : starting from a star tree, NJ iteratively agglomerates external pairs of copies so as to minimize the tree length at each step.

Assuming that we have consistent evolutionary distance estimators which converge towards the true distance as the length of the sequences increases, the ME criterion combined with ordinary least-squares (OLS) is statistically consistent [14,15], as opposed to the parsimony criterion [16]. Statistical consistency is an essential property in phylogenetic reconstruction, since it ensures that, for the given method, the probability of recovering the correct topology increases with the amount of data.

In this section, we introduce a new recurrence equation for estimating the length of a tree topology using OLS, given a matrix of pairwise evolutionary distances between copies. As we shall see, this equation forms the basis of our exact reconstruction algorithm.

3.2 Notation

Δ is a matrix of pairwise evolutionary distances between copies, and δ_{ij} is the distance in Δ between copy i and copy j ; \mathcal{T} is an unrooted tree topology, and T represents a valued tree with topology \mathcal{T} . T induces a matrix of pairwise tree distances between copies, which we denote Δ^T . In this matrix, δ_{ij}^T denotes the length of the tree path linking copy i and copy j . The sum of the branch lengths of T is denoted as $L(T)$. As shown in Figure 2, we consider in the rest of this section that T is composed of three non-intersecting subtrees A , B and C . These subtrees are linked together by three branches whose lengths are a , b and c . A is the union of two subtrees A_1 and A_2 , and in turn A_1 is the union of two subtrees A_{11} and A_{12} . Two branches with lengths a_1 and a_2 link the root of A to the roots of A_1 and A_2 , respectively. In the remainder of this paper, we call R the subset of leaves that do not belong to A (*i.e.* $R = B \cup C$).

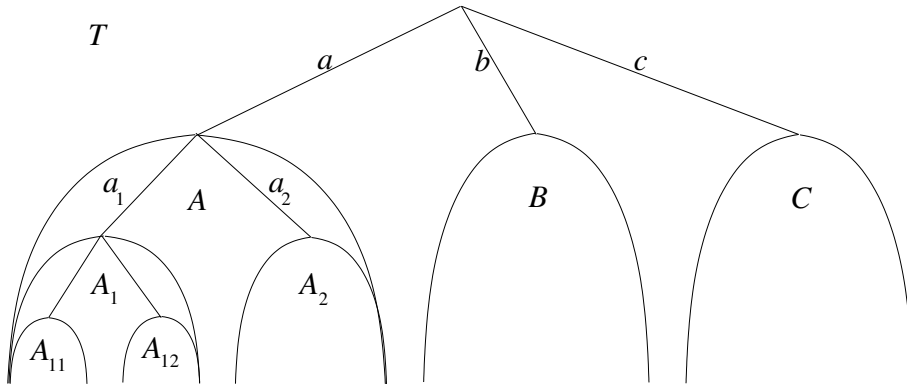


Figure 2. unrooted tree T , composed of three subtrees A , B and C .

Let X be any subtree of T , and \bar{X} be the average distance in T between the root of X and its leaves. Δ_{XY} and Δ_{XY}^T are the average distances between the leaves of two non-intersecting subtrees X and Y , in the distance matrices Δ and Δ^T , respectively :

$$\Delta_{XY} = \frac{1}{|X||Y|} \sum_{i \in X, j \in Y} \delta_{ij}$$

$$\Delta_{XY}^T = \frac{1}{|X||Y|} \sum_{i \in X, j \in Y} \delta_{ij}^T.$$

Given a topology \mathcal{T} and a distance matrix Δ , the OLS branch length estimation of T is obtained by minimizing the following sum of squares:

$$\sum_{i,j \in T} (\delta_{ij}^T - \delta_{ij})^2.$$

3.3 OLS tree length expression

Theorem : Let the branches of T be estimated by OLS. Then :

$$L(T) = (L(A) - \bar{A}) + (L(B) - \bar{B}) + (L(C) - \bar{C}) + \frac{1}{2}(\Delta_{AB} + \Delta_{AC} + \Delta_{BC}). \quad (1)$$

Moreover, $(L(A) - \bar{A})$ is recursively obtained in the following way :

- (a) if A is a leaf, then $(L(A) - \bar{A}) = 0$,
- (b) otherwise, $(L(A) - \bar{A})$ is given by :

$$(L(A) - \bar{A}) = (L(A_1) - \bar{A}_1) + (L(A_2) - \bar{A}_2) + \frac{1}{2}\Delta_{A_1A_2} + \frac{1}{2}\left(\frac{|A_2| - |A_1|}{|A|}\right)\Delta_{A_1R} + \frac{1}{2}\left(\frac{|A_1| - |A_2|}{|A|}\right)\Delta_{A_2R}, \quad (2)$$

and the same applies to $(L(B) - \bar{B})$ and $(L(C) - \bar{C})$, by symmetry.

Proof : Using Figure 2, we see that :

$$L(T) = L(A) + L(B) + L(C) + a + b + c. \quad (3)$$

It has been shown that the average distance between two non-intersecting subtrees X and Y is preserved between Δ and Δ^T , when these subtrees are adjacent to a common ternary node (*i.e.* A and B, A and C or B and C in Figure 2), and when branch lengths of T are estimated by OLS [20,21]. Using this property, Δ_{AB} , Δ_{AC} and Δ_{BC} can be expressed in the following way :

$$\begin{aligned} (i) \quad \Delta_{AB} &= \bar{A} + a + b + \bar{B}, \\ (ii) \quad \Delta_{AC} &= \bar{A} + a + c + \bar{C}, \\ (iii) \quad \Delta_{BC} &= \bar{B} + b + c + \bar{C}, \end{aligned} \quad (4)$$

and Equation (1) is obtained by combining Equations (3) and (4). Identically, the length of A is equal to the sum of its branch lengths :

$$L(A) = L(A_1) + L(A_2) + a_1 + a_2, \quad (5)$$

while \bar{A} is given by :

$$\bar{A} = \frac{|A_1|}{|A|}(a_1 + \bar{A}_1) + \frac{|A_2|}{|A|}(a_2 + \bar{A}_2). \quad (6)$$

a_1 (and a_2 by symmetry) is obtained by replacing Δ_{AB} , Δ_{AC} , Δ_{BC} and a by $\Delta_{A_1A_2}$, Δ_{A_1R} , Δ_{A_2R} and a_1 in Equation (4), respectively. If we add Equations (4-i) and (4-ii), and replace $b + c$ by its expression obtained from (4-iii), we obtain :

$$\begin{aligned} a_1 &= \frac{1}{2}\Delta_{A_1A_2} + \frac{1}{2}\Delta_{A_1R} - \frac{1}{2}\Delta_{A_2R} - \bar{A}_1 \\ a_2 &= \frac{1}{2}\Delta_{A_1A_2} + \frac{1}{2}\Delta_{A_2R} - \frac{1}{2}\Delta_{A_1R} - \bar{A}_2 \end{aligned} \quad (7)$$

Equation (2) is finally obtained by subtracting (6) to (5), and replacing a_1 and a_2 by their analytical expression (7), while the equality $(L(A) - \bar{A}) = 0$, if A is a leaf, is a direct consequence of the definitions.

3.4 Properties

In Equation (1), $(L(A) - \bar{A})$, $(L(B) - \bar{B})$ and $(L(C) - \bar{C})$ only depend on the structure of subtrees A , B and C , respectively. Indeed, Equation (7) clearly shows that the branch length a_1 depends on the copies in subtrees A_1 , A_2 and $R = B \cup C$, but not on the structure of R (*i.e.* the content of B and C). The same applies with a_2 . Identically, this property is valid for branch length a_{11} , which depends on the copies in subtrees A_{11} , A_{12} and $R' = A_2 \cup R$, but not on the structure of R' , and therefore not on the structure of R . It can be established in this way that none of the branch lengths in A depends on the structure of R . Therefore, to compute $L(T)$, we independently compute the values for $(L(A) - \bar{A})$, $(L(B) - \bar{B})$ and $(L(C) - \bar{C})$, and then apply Equation (1).

For the same reasons, $(L(A_1) - \bar{A}_1)$ and $(L(A_2) - \bar{A}_2)$ only depend on the structure of A_1 and A_2 , respectively. Therefore, to compute $(L(A) - \bar{A})$, we independently compute the values for $(L(A_1) - \bar{A}_1)$ and $(L(A_2) - \bar{A}_2)$, and then apply Equation (2).

4 Reconstructing optimal single copy duplication trees

4.1 Basic algorithm

The above recurrence equations enable to calculate the OLS length of given unrooted tree topology, given a matrix of pairwise distances. In this section, we seek the duplication tree whose length is minimum, when estimated with the above equations. Equation (1) consists of four independent terms $(L(A) - \bar{A})$, $(L(B) - \bar{B})$, $(L(C) - \bar{C})$ and the remaining term. As we said above, $(L(A) - \bar{A})$, $(L(B) - \bar{B})$ and $(L(C) - \bar{C})$ only depend on the structure of subtrees A , B and C , respectively, while the remaining term consists of average distances, and therefore does not depend on the structure of A , B and C . To minimize Equation (1), we adopt a divisive strategy, which consists first in partitioning the whole set of copies into three subsets A , B and C , then in computing the structure which minimizes $(L(X) - \bar{X})$ for each of these subsets, and finally in applying Equation (1). The optimal tree is given by the optimal partitioning. Identically, to obtain the optimal structure for A , Equation (2) shows that we need to evaluate every partitioning of A into A_1 and A_2 , then to compute the structure for A_1 and A_2 which minimizes $(L(X) - \bar{X})$ and finally to select the partitioning which minimizes Equation (2).

Although used in some divisive clustering methods [22], this strategy cannot be used to reconstruct optimal phylogenies when n is large, since the number of combinations of subsets is exponential. This is different with single copy duplication trees, since we only have to evaluate combinations of (two or three) adjacent intervals, and the number of combinations is polynomial.

Let S and M be two $n \times n$ matrices. $S_{p,q}$ represents the minimal value of $(L(X_{p,q}) - \bar{X}_{p,q})$ where $X_{p,q}$ is any subtree with leaves in $[p, q]$, while $M_{p,q}$ represents the value of m for which $S_{p,q}$ is minimal (see below). Let $X_{\bar{p},\bar{q}}$ represent the subset of copies that do not belong to $X_{p,q}$ (*i.e.* $X_{\bar{p},\bar{q}} = X_{1,p-1} \cup X_{q+1,n}$). Starting from an interval $[1, n]$ representing the n copies, the reconstruction algorithm for single copy duplication trees necessitates the three following steps :

(a) the first step consists in using Equation (2) to calculate $S_{p,q}$ for a growing interval $X_{p,q}$ of $[1, n]$, until $q - p = n - 3$. Computing $S_{p,q}$ requires evaluating the combination of every couple of adjacent intervals $X_{p,m}$ and $X_{m+1,q}$, with m varying from p to $q - 1$. Therefore, using Equation (2), $S_{p,q}$ is given by :

$$S_{p,q} = \min_{p \leq m \leq q-1} \left[S_{p,m} + S_{m+1,q} + \frac{1}{2} \Delta_{X_{p,m} X_{m+1,q}} + \frac{1}{2} \left(\frac{p+q-2m-1}{q-p+1} \right) \Delta_{X_{p,m} X_{\bar{p},\bar{q}}} + \frac{1}{2} \left(\frac{2m-p-q+1}{q-p+1} \right) \Delta_{X_{m+1,q} X_{\bar{p},\bar{q}}} \right], \quad (8)$$

while $M_{p,q}$ is the value of m minimizing the above expression. Moreover, we have $S_{p,p} = 0$ for $1 \leq p \leq n$.

(b) the second step consists in using Equation (1) to search for the set of three adjacent intervals X_{1,m_1} , X_{m_1+1,m_2} and $X_{m_2+1,n}$ which minimizes $L(T)$.

(c) in the third step, the complete tree topology is recovered by stepping back through the optimal intervals stored in M .

Algorithm 1 Single copy duplication tree reconstruction algorithm

```

 $S \leftarrow n \times n$  matrix
 $M \leftarrow n \times n$  matrix
for  $l$  from 1 to  $n - 3$  do
  for  $i$  from 1 to  $n - l$  do
    compute  $S_{i,i+l}$  and  $M_{i,i+l}$  using Equations (8)
  end for
end for
 $L^*(T) \leftarrow \infty$ 
for  $m_1$  from 1 to  $n - 2$  do
  for  $m_2$  from  $m_1 + 1$  to  $n - 1$  do
    compute  $L(T)$  for  $X_{1,m_1}$ ,  $X_{m_1+1,m_2}$ ,  $X_{m_2+1,n}$  using Equation (1) and  $S$ 
    if  $L(T) < L^*(T)$  then
       $L^*(T) \leftarrow L(T)$ ,  $m_1^* \leftarrow m_1$ ,  $m_2^* \leftarrow m_2$ 
    end if
  end for
end for
recursively create  $T$  using the values in  $M$ , starting from  $M_{1,m_1^*}$ ,  $M_{m_1^*+1,m_2^*}$  and  $M_{m_2^*+1,n}$ 

```

This algorithm is summarized above. The number of intervals which need to be evaluated during the first step is in $O(n^2)$. Evaluating a single interval using Equation (8) necessitates the evaluation of $O(n)$ combinations of adjacent sub-intervals. Evaluating a single combination requires the average distances between the $X_{p,m}$, $X_{m+1,q}$ and $X_{\overline{p,q}}$ subsets to be computed, and necessitates $O(n^2)$ time. Therefore, the total time complexity of the first step is $O(n^5)$. As we show in the next section, the time required to compute a single combination of adjacent sub-intervals can be lowered to $O(1)$. When using this refinement, the total time complexity of the first step is lowered to $O(n^3)$.

In the second step, we evaluate every possible combination of three adjacent intervals X_{1,m_1} , X_{m_1+1,m_2} and $X_{m_2+1,n}$. Therefore, the number of combinations that need to be tested in the second step is also in $O(n^2)$. As in the previous step, evaluating a single combination requires average distances between intervals to be computed. Therefore, the time complexity of the second step is $O(n^4)$, and can be lowered to $O(n^2)$ when using another algorithmic refinement, as described below. Since the last step is only a depth-first tree traversal, it requires $O(n)$ time, so the total time complexity is $O(n^5)$ in the above “basic” description of our algorithm, and $O(n^3)$ using the following refinement.

4.2 $O(1)$ computation of the average distances between intervals

Consider an interval $X_{p,q}$, composed of two adjacent intervals $X_{p,m}$ and $X_{m+1,q}$. As shown above, the calculation of $(L(X_{p,q}) - \overline{X_{p,q}})$ requires three average distances $\Delta_{X_{p,m}X_{m+1,q}}$, $\Delta_{X_{p,m}X_{\overline{p,q}}}$, $\Delta_{X_{m+1,q}X_{\overline{p,q}}}$ to be computed. In this section, we describe a refinement whose basic idea is the following : since we are considering growing intervals, most of the work needed for calculating these average distances for intervals of current size l has already been done when intervals of size $(l - 1)$ and $(l - 2)$ were considered. We show below that calculating these average distances can be done in $O(1)$ time, using a memorization scheme, thus lowering the complexity of our algorithm to $O(n^3)$.

$O(1)$ computation of $\Delta_{X_{p,m} X_{m+1,q}}$ It is easy to show that the average distance between two intervals $X_{p,m}$ and $X_{m+1,q}$ can be expressed as follows :

$$\Delta_{X_{p,m} X_{m+1,q}} = \frac{1}{(m-p+1)(q-m)} \begin{pmatrix} (m-p+1)(q-m-1)\Delta_{X_{p,m} X_{m+1,q-1}} \\ + (m-p)(q-m)\Delta_{X_{p+1,m} X_{m+1,q}} \\ - (m-p)(q-m-1)\Delta_{X_{p+1,m} X_{m+1,q-1}} \\ + \delta_{p,q} \end{pmatrix}. \quad (9)$$

Let $l = q - p + 1$ be the length of the current interval. In Equation (9), both $\Delta_{X_{p,m} X_{m+1,q-1}}$ and $\Delta_{X_{p+1,m} X_{m+1,q}}$ have been calculated when we considered intervals of length $(l - 1)$, while $\Delta_{X_{p+1,m} X_{m+1,q-1}}$ has been calculated when we considered intervals of length $(l - 2)$. Therefore, calculating $\Delta_{X_{p,m} X_{m+1,q}}$ can be done in $O(1)$ provided the required average distances have been stored when we considered both previous interval lengths. This memorization procedure requires $O(n^2)$ space, since we only need to store average distances for intervals with lengths $l - 1$ and $l - 2$.

$O(1)$ computation of $\Delta_{X_{p,m} X_{p,q}}$ and $\Delta_{X_{m+1,q} X_{p,q}}$ Equations (2) and (8) are identical, except that the A_1 , A_2 and R notation in Equation (2) is replaced with $X_{p,m}$, $X_{m+1,q}$ and $X_{p,q}$ in Equation (8), respectively. In this section, we use A_1 , A_2 and R , for simplicity. In Equation (2), computing $(L(A) - \bar{A})$ requires both $\Delta_{A_1 R}$ and $\Delta_{A_2 R}$ to be known, where $A = A_1 \cup A_2$ and R is the set of copies which do not belong to A_1 or A_2 , *i.e.* $R = T - (A_1 \cup A_2)$. It is then easy to show that $\Delta_{A_1(T-A_1)}$ is given by the following equation:

$$\Delta_{A_1(T-A_1)} = \frac{1}{n - |A_1|} (|A_2| \Delta_{A_1 A_2} + (n - |A_1| - |A_2|) \Delta_{A_1 R}),$$

which yields:

$$\Delta_{A_1 R} = \frac{1}{(n - |A_1| - |A_2|)} \left((n - |A_1|) \Delta_{A_1(T-A_1)} - |A_2| \Delta_{A_1 A_2} \right), \quad (10)$$

and $\Delta_{A_2 R}$ is obtained by symmetry. In the previous section, we showed that $\Delta_{A_1 A_2}$ can be calculated in $O(1)$. Therefore, we only need to know the average distance between A_1 and $(T - A_1)$ to compute $\Delta_{A_1 R}$ using Equation (10). The same applies to A_2 . The refinement we introduce here consists in precalculating and storing $\Delta_{A_1(T-A_1)}$ for every possible interval A_1 , so that $\Delta_{A_1 R}$ can be calculated in $O(1)$ time.

Assume that $\Delta_{Z(T-Z)}$ has been calculated for all intervals Z with length $(l - 1)$. Let now Z be an interval with length l . Let z be the leftmost (or rightmost) copy of Z , which ‘‘jumps’’ at the current step from $(T - (Z - \{z\}))$ to Z . $Z - \{z\}$ has length $(l - 1)$ and $\Delta_{(Z-\{z\})(T-(Z-\{z\}))}$ has already been computed. $\Delta_{Z(T-Z)}$ can be computed in $O(n)$ time using the following equation:

$$\Delta_{Z(T-Z)} = \frac{1}{|Z|(n - |Z|)} \left((|Z| - 1)(n - |Z| + 1) \Delta_{(Z-\{z\})(T-(Z-\{z\}))} - \sum_{y \in (Z-\{z\})} \delta_{yz} + \sum_{y \in (T-Z)} \delta_{yz} \right).$$

$O(n^2)$ intervals need to be considered, and each of them is evaluated in $O(n)$ time. Therefore, precomputing every $\Delta_{Z(T-Z)}$ is done in $O(n^3)$ time, and the space required to store the resulting values is in $O(n^2)$.

$O(1)$ computation of Δ_{AB} , Δ_{AC} , Δ_{BC} The second step of our reconstruction algorithm requires the values of Δ_{AB} , Δ_{AC} , and Δ_{BC} within Equation (1) to be calculated. It is easy to show that these values can directly be calculated in $O(1)$ from $\Delta_{A(B \cup C)}$, $\Delta_{B(A \cup C)}$ and $\Delta_{C(A \cup B)}$, using the following set of equations :

$$\begin{aligned}\Delta_{A(B \cup C)} &= \Delta_{A(T-A)} = \frac{1}{|B|+|C|} (|B| \Delta_{AB} + |C| \Delta_{AC}), \\ \Delta_{B(A \cup C)} &= \Delta_{B(T-B)} = \frac{1}{|A|+|C|} (|A| \Delta_{AB} + |C| \Delta_{BC}), \\ \Delta_{C(A \cup B)} &= \Delta_{C(T-C)} = \frac{1}{|A|+|B|} (|A| \Delta_{AC} + |B| \Delta_{BC}).\end{aligned}$$

When solving the above linear system, we obtain the following expressions for Δ_{AB} :

$$\Delta_{AB} = \frac{1}{2|A||B|} \left(|A| [|B| + |C|] \Delta_{A(T-A)} + |B| [|A| + |C|] \Delta_{B(T-B)} - |C| [|A| + |B|] \Delta_{C(T-C)} \right),$$

and Δ_{AC} , Δ_{BC} are obtained by symmetry. Since every possible $\Delta_{Z(T-Z)}$ has been computed and stored during the first step, Δ_{AB} , Δ_{AC} , and Δ_{BC} are calculated from the above expression in $O(1)$.

5 Conclusion

In this paper, we presented an exact algorithm to reconstruct single copy duplication trees from a matrix of evolutionary distances between tandemly repeated sequences, using the minimum evolution criterion. This algorithm is based on a new recurrence equation for ordinary least-squares estimation of tree length. We showed that, using a simple memorization scheme and a dynamic programming approach, computing the optimal single copy duplication tree can then be done in $O(n^3)$ time and $O(n^2)$ space. It would now be relevant to compare our algorithm with some of the classical approaches, in terms of topological accuracy. Indeed, heuristic methods such as NJ [19] or DTSCORE [23] often do well in practice. Moreover, NJ could easily be adapted to single copy duplication tree reconstruction by only agglomerating adjacent pairs of taxa. But an exact algorithm such as ours has performance guaranty and will then avoid some possible (even rare) shortcomings that would trap heuristic approaches into local minima. So in phylogenetic reconstruction as in other domains, the search for polynomiality remains of importance. However, the parsimony version of the single duplication tree problem was shown to be NP-hard and, in fact, most evolutionary tree reconstruction problems (except combinatorial methods which only reconstruct partially resolved trees, *e.g.* [24], or some instances of the perfect phylogeny problem from character data [25]) are NP-hard and require the use of heuristics. It follows that the problem described in this paper is one of the first to accept an exact and polynomial solution, in the field of evolutionary tree reconstruction. So another direction for further research is to extend (if possible) our results to multiple duplications and to other distance criteria, such as weighted least-squares [26,27], or to demonstrate the NP-hardness of these tasks.

References

1. Ohno, S.: Evolution by gene duplication. Springer Verlag, New York (1970)
2. Smith, G.: Evolution of repeated dna sequences by unequal crossover. *Science* **191** (1976) 528–535
3. Fitch, W.: Phylogenies constrained by cross-over process as illustrated by human hemoglobins in a thirteen-cycle, eleven amino-acid repeat in human apolipoprotein A-I. *Genetics* **86** (1977) 623–644
4. Jeffreys, A., Harris, S.: Processes of gene duplication. *Nature* **296** (1981) 9–10
5. Elemento, O., Gascuel, O., Lefranc, M.P.: Reconstruction de l’histoire de duplication de gènes répétés en tandem. In: Actes des Journées Ouvertes Biologie Informatique Mathématiques. (2001) 9–11
6. Elemento, O., Gascuel, O., Lefranc, M.P.: Reconstructing the duplication history of tandemly repeated genes. *Molecular Biological Evolution* **19** (2002) 278–288
7. Benson, G., Dong, L.: Reconstructing the duplication history of a tandem repeat. In Lengauer, T., Schneider, R., Bork, P., Brutlag, D., Glasgow, J., Mewes, H.W., Zimmer, R., eds.: Proceedings of Intelligent Systems in Molecular Biology ISMB’99. (1999) 44–53

8. Tang, M., Waterman, M., Yooseph, S.: Zinc finger gene clusters and tandem gene duplication. In El-Mabrouk, N., Lengauer, T., Sankoff, D., eds.: Proceedings of RECOMB 2001. (2001) 297–304
9. Tang, M., Waterman, M., Yooseph, S.: Zinc finger gene clusters and tandem gene duplication. *Journal of Computational Biology* **9** (2002) 429–446
10. Jaitly, D., Kearney, P., Lin, G., Ma, B.: Methods for reconstructing the history of tandem repeats and their application to the human genome. *Journal of Computer and System Sciences* **65** (2002) 494–507.
11. Zhang, J., Nei, M.: Evolution of antennapedia-class homeobox genes. *Genetics* **142** (1996) 295–303
12. Wang, L., Gusfield, D.: Improved approximation algorithms for tree alignment. *Journal of Algorithms* **25** (1997) 255–273
13. Kidd, K., Sgaramella-Zonta, L.: Phylogenetic analysis: concepts and methods. *American Journal of Human Genetics* **23** (1971) 235–252
14. Rzhetsky, A., Nei, M.: Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Molecular Biological Evolution* **10** (1993) 173–1095
15. Denis, F., Gascuel, O.: On the consistency of the minimum evolution principle of phylogenetic inference. *Discrete Applied Mathematics* (2002) In press.
16. Felsenstein, J.: Cases in which parsimony or compatibility methods will be positively misleading. *Systematic Zoology* **27** (1978) 401–410
17. Vardi, I.: *Computational Recreations in Mathematica*. Addison-Wesley (1991)
18. Day, W.: Computational complexity of inferring phylogenies from dissimilarity matrices. *Bulletin of Mathematical Biology* **49** (1987) 461–467
19. Saitou, N., Nei, M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution* **4** (1987) 406–425
20. Vach, W.: Least-squares approximation of additive trees. In Opitz, O., ed.: *Conceptual and Numerical Analysis of Data*, Heidelberg, Springer (1989) 230–238
21. Gascuel, O.: Concerning the NJ algorithm and its unweighted version, UNJ. In Mirkin, B., McMorris, F., Roberts, F., Rzhetsky, A., eds.: *Mathematical Hierarchies and Biology*. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Amer. Math. Society, Providence (1997) 149–170
22. Barthelemy, J., Guénoche, A.: *Trees and proximity representations*. Wiley and Sons (1991)
23. Elemento, O., Gascuel, O.: A fast and accurate distance-based algorithm to reconstruct tandem duplicatin trees. *Bioinformatics* **18** (2002) S92–S99 Proceedings of European Conference on Computational Biology (ECCB2002).
24. Bryant, D., Berry, V.: A structured family of clustering and tree reconstruction methods. *Advances in Applied Mathematics* **27** (2001) 705–732
25. Kannan, S., Warnow, T.: A fast algorithm for the computation and enumeration of perfect phylogenies when the number of character states is fixed. *SIAM J. Computing* **26** (1997) 1749–1763
26. Fitch, W., Margoliash, E.: Construction of phylogenetic trees. *Science* **155** (1967) 279–284
27. Felsenstein, J.: An alternating least squares approach to inferring phylogenies from pairwise distances. *Systematic Biology* **46** (1997) 101–111