Università di Pisa
Dipartimento di Informatica

Technical Report: TR-10-01

Lower Bounds on the Rotation Distance of Binary Trees

Fabrizio Luccio Antonio Mesa Enriquez Linda Pagli

January 4, 2010

address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. tel: +39 050 2212700 fax: +39 050 2212726

Lower Bounds on the Rotation Distance of Binary Trees

Fabrizio Luccio Antonio Mesa Enriquez Linda Pagli

January 4, 2010

Abstract

The *rotation distance* $d(S,T)$ between two binary trees S, T of n vertices is the minimum number of rotations to transform S into T. While it is known that $d(S,T) \leq 2n-6$, a well known conjecture states that there are trees for which this bound is sharp for any value of $n \geq 11$. We are unable to prove the conjecture, but we give here some simple criteria for lower bound evaluation, leading for example to individuate some "regular" tree structures for which $d(S,T) = 3n/2 - O(1)$, or $d(S,T) = 5n/3 - O(1)$.

Keywords: Rotation distance, Lower bound, Binary tree, Design of algorithms.

1 The problem

Consider two rooted binary trees S and T of n vertices, simply called *trees* in the following. The vertices of both trees may be numbered from 1 to n in *infix order*, so that all the integers of the left (respectively, right) subtree of each vertex v are smaller (respectively, greater) than the integer of v. Vertices will be identified by these numbers. A *rotation* of two adjacent vertices of a tree is a local change of the vertex positions preserving the infix order of the numbering (see next section). The *rotation distance* $d(S,T)$ between S and T is the minimum number of rotations by which S can be transformed into T. This concept was introduced by Culik and Wood [1]. Sleator, Tarjan, and Thurston [9] proved that $d(S,T) \leq 2n-6$ for any pair S, T, with $n \geq 11$. Mäkinen [6] showed that slightly weaker results can be obtained by an elementary procedure, and Luccio and Pagli [5] gave a simple combinatorial proof for the upper bound of [9]. More important, by a deep geometric argument the authors of [9] proved that the upper bound $2n-6$ is sharp for (ineffectively) large values of n, and conjectured that is sharp for any $n \geq 11$. No efficient algorithm is known to compute $d(S,T)$, but estimates were given by Pallo [7] and Rogers [8]. It is not even known if the problem is NP-hard.

In this note we prove some elementary properties of binary trees leading to the formulation of lower bound criteria for $d(S,T)$, and show that such criteria are significant for some particular families of trees. The same results may give hints on the construction of a "good" rotation algorithm.

2 Lower bound arguments

A rotation $\mathrm{rot}(x,y)$ is defined for a pair of parent-child vertices x, y, called the *parameters* of rot. For $y < x$ (i.e., y is the left child of x), $\mathrm{rot}(x,y)$ raises y to the place of x while x becomes the right child of y, the original right subtree of y becomes the left subtree of x, and the rest of the tree remains unchanged. This is called a *right* rotation, see figure 1. For $y > x$ (i.e., y is the right child of x), $\mathrm{rot}(x,y)$ is symmetrical and is called a *left* rotation. Two consecutive rotations $\mathrm{rot}(x,y)$, $\mathrm{rot}(y,x)$ leave the tree unchanged. This also implies that $d(S,T) = d(T,S)$ for two arbitrary trees S, T.

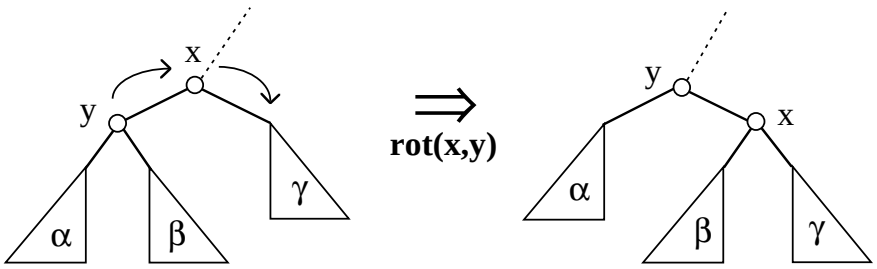

Figure 1: A right rotation ($y < x$).

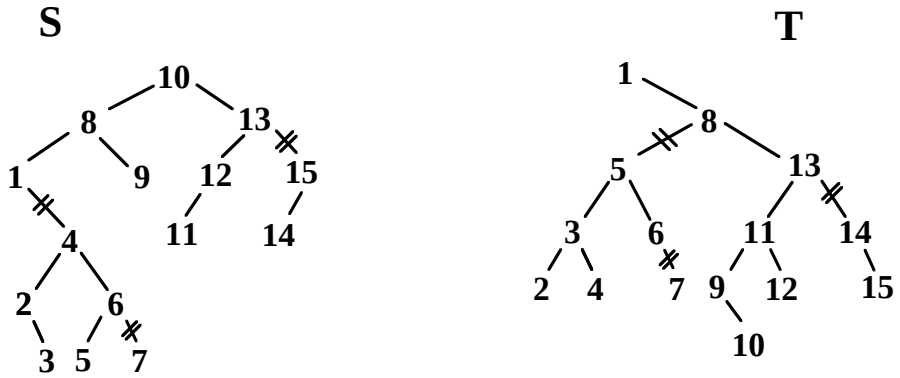

Figure 2: Partitioning S, T into two forests F_S, F_T.

We now establish some notation and review some simple properties of binary trees. For any vertex x of a tree S, let SUB(x) be the subtree of S rooted at x; and let $ld(x)$, $rd(x)$ be the number of left and right descendants of x, i.e. the number of vertices in the subtrees rooted at the left-child and at the right-child of x. We have $0 \leq ld(x) \leq x-1$, $0 \leq rd(x) \leq n-x$. If needed a subscript S will be added to the notation to denote tree S.

An immediate consequence of the infix order is that the vertices of SUB(x) form a closed non empty interval INT(x) = $[x - ld(x), x + rd(x)]$. For the trees S, T of figure 2 we have $\text{INT}_S(4)$ = [2,7], $\text{INT}_S(7)$ = [7,7], $\text{INT}_T(8)$ = [2,15]. The interval INT(x) (in fact, the pair of values $ld(x)$, $rd(x)$) identifies the vertices of SUB(x) but not its shape. Given two trees S, T, if $S = T$ we obviously have $\text{INT}_S(x) = \text{INT}_T(x)$ for all x. If $S \neq T$, $d(S,T)$ is the minimum number of rotations needed to attain the above condition.

Let $x \updownarrow y$ indicate that vertices x and y lie on the same path from the root to a leaf, and let $x \downarrow y$ indicate that $x \updownarrow y$ and x is closer to the root than y. We have:

Fact 1 *For each vertex x, $1 \leq x \leq n-1$, we have $x \updownarrow x{+}1$.*
Proof. Let the tree be built as a binary search tree with the node numbers treated as keys. If vertex x has entered the tree before $x{+}1$, then $x{+}1$ follows the same path from the root to the position of x before proceeding further down in the path to a leaf. In this case we have $x \downarrow x{+}1$. The same reasoning applies if $x{+}1$ enters the tree before x, and we have $x{+}1 \downarrow x$. *Q.E.D.*

Consider two vertices x, y respectively in S, T, with $x \neq \text{root}(S)$, $y \neq \text{root}(T)$, and $\text{INT}_S(x) =$

$\mathrm{INT}_T(y)$. The corresponding subtrees $\mathrm{SUB}_S(x)$, $\mathrm{SUB}_T(y)$ are said to be *equivalent*. In fact they have the same vertices (specified in the intervals), although they may have different roots if $x \neq y$. For example in figure 2 we have $\mathrm{INT}_S(4) = \mathrm{INT}_T(5) =$ [2,7], hence $\mathrm{SUB}_S(4)$ and $\mathrm{SUB}_T(5)$ are equivalent. Similary $\mathrm{SUB}_S(15)$, $\mathrm{SUB}_T(14)$ are equivalent, and $\mathrm{SUB}_S(7)$, $\mathrm{SUB}_T(7)$ are equivalent. Consider now a pair of vertices z, x, where $\mathrm{INT}(x) =$ [a,b] and x is a child of z. An immediate consequence of Fact 1 is that if $z < x$ then $z = a - 1$, if $z > x$ then $z = b + 1$. In figure 2, for tree S and vertex $x = 4$ we have $\mathrm{INT}_S(4) =$ [2,7] and the parent of $x = 4$ is $z = 1 < 4$: in fact $z = a - 1$. In the equivalent subtree we have $\mathrm{INT}_T(5) =$ [2,7] and the parent of x is $z = 8 > 5$: in fact $z = b + 1$.

Using the above property we can find all the pairs of equivalent subtrees in S, T in linear time with the algorithm EQSUB reported in Table 1. Upon exit from the algorithm, stack A contains all the pairs of roots of the equivalent subtrees in S, T. For the example of figure 2, A will contain <15,14>, <7,7>, <4,5>. From an elementary analysis of Algorithm EQSUB we have:

Fact 2 *Given two trees S, T, all the pairs of equivalent subtrees can be determined in $O(n)$ time.*

Once all the pairs of equivalent subtrees of S, T have been determined, the two trees can be transformed into two forests $F_S = \{S_1, ..., S_k\}$, $F_T = \{T_1, ..., T_k\}$ by cutting all the edges connecting the equivalent subtrees to the rest of S and T (see figure 2). A (non immediate) consequence of the geometric argument developed in [9] is that a shortest sequence of rotations to transform S into T can be divided into k independent subsequences to transform each S_i into T_i. In particular if S_i and T_i consist of a single vertex x (e.g., x is a leaf in both S and T), then x will never be rotated in a shortest sequence. In figure 2, S and T are partitioned into four independent trees respectively containing the vertices 2 to 6, 7, 1 together with 8 to 13, and 14-15. Due to Fact 2 this partition is obtained here in linear time, while the corresponding result for the technique of [9] was obtained in quadratic time in [4].

```
algorithm EQSUB(S, T)
    let A be an empty stack of pairs;
    for (x ∈ {1, ..., n}) compute {INT_S(x); INT_T(x)};
    for (x ∈ {1, ..., n} and x ≠ root(S)) {
        if (INT_S(x) = INT_T(x)) < x, x > → A
        else {
            let INT_S(x) = [a,b];
            if (a > 1) {
                y = rightchild(a-1) in T;
                if (INT_T(y) = [a,b]) < x, y > → A; }
            if (b < n) {
                y = leftchild(b+1) in T;
                if (INT_T(y) = [a,b]) < x, y > → A; } } }
```

Table 1. Computing equivalent subtrees in linear time.

We can now prove our first lower bound result, namely:

Lemma 1 *$d(S, T) \geq e$, where e is the total number of edges in the forest F_S, and the value of e can be computed in $O(n)$ time.*

Proof. In a rotation rot(x,y) only the intervals $\mathrm{INT}(x)$, $\mathrm{INT}(y)$ change (see figure 1). However the value of $\mathrm{INT}(y)$ after the rotation is equal to the value of $\mathrm{INT}(x)$ before the rotation, while a

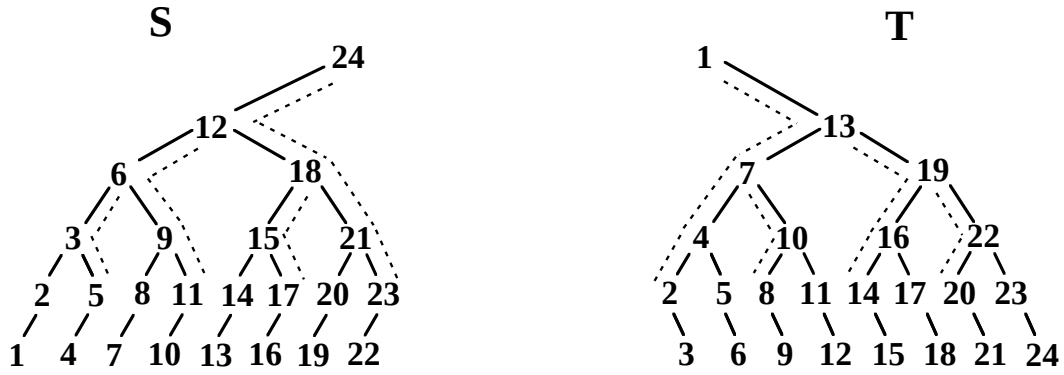

Figure 3: Two trees with the maximum number of inversions. For trees with this shape we have $d(S,T) = 5n/3-\mathrm{O}(1)$ (see section 3).

new value arises for INT(x) after the rotation. So in the set of all intervals of the tree exactly one value changes after a rotation. If the sets of intervals relative to S and T have c different elements we then have $d(S,T) \geq c$. Note now that the roots of S, T correspond to the same interval [1,n], and any two equivalent subtrees in S, T correspond to two equal intervals. Therefore if the trees can be decomposed into two forests F_S, F_T of k pairwise equivalent subtrees, the lists of intervals of S and T differ for $c = n - 1 - k$ intervals. It can be immediately seen that this value equals e, the number of edges in F_S (or F_T, see figure 2). As a consequence of Fact 2, the value of e can be computed in linear time. *Q.E.D.*

For the trees of figure 2 we have $d(S,T) \geq 5 + 5 + 1 = 11$. We also have:

Corollary 1 *If $F_S = S$, $F_T = T$ then $d(S,T) \geq n - 1$.*

This is the case of the trees in figure 3, for which we have $d(S,T) \geq 23$.

The vertices x, x+1 are called *consecutive*. Given two trees S, T, if $x \downarrow x$+1 in S (respectively, in T), and x+1 $\downarrow x$ in T (respectively, in S), then x, x+1 are said to form an *inversion*. $I(S,T)$ denotes the number of inversions in S, T. We have:

Fact 3 *If x, x+1 form an inversion in S, T, then any rotation algorithm to transform S into T must include a rotation with parameters x, x+1.*

Proof. Let $x \downarrow x$+1 in S; let α, β be the left and right subtrees of x; and let γ be the rest of S deprived of x and all its descendants. Note that $\pi(x,y)$ lies in α. Any rotation with both parameters in α, or in β, or in γ, maintains $x \downarrow x$+1. Then a rotation rot(x, x+1) is needed to invert the order of the two vertices. The same reasoning applies if x+1 $\downarrow x$ in S. *Q.E.D.*

Therefore we immediately have:

Lemma 2 *$d(S,T) \geq I(S,T)$.*

For example the two trees of figure 3 have $n - 1 = 23$ inversions, then $d(S,T) \geq 23$. Note that this bound was already derived from Corollary 1.

3 Sharpening the bounds

Lemmas 1 and 2 give a basis for fixing a lower bound to $d(S,T)$, but the emerging value is at most $n-1$, far from the uper bound of $2n-1$ that applies to all trees. Two subtler arguments, however, may allow to sharpen such a lower bound substantially.

The technique of Lemma 1 is improved by comparing the intervals $\mathrm{INT}_S(x)$, $\mathrm{INT}_T(x)$ of the same vertex x in the two trees. In fact, to transform S into T such intervals must become equal for all vertices. As already noted in the proof of Lemma 1, in a rotation rot(x,y) only the intervals of vertices x and y change (see figure 1). Let $\mathrm{INT}(x)=[l_x, r_x]$, $\mathrm{INT}(y)=[l_y, r_y]$ (another subscript may be added to indicate the tree). In a right rotation ($y < x$) the values of l_x and r_y increase while the values of r_x and l_y keep their values. In a left rotation ($y > x$) the values of r_x and l_y decrease while the values of l_x and r_y keep their values.

Given two trees S, T, let L_+, L_-, R_+, R_- respectively denote the number of vetices x such that $l_{x,S} > l_{x,T}$, or $l_{x,S} < l_{x,T}$, or $r_{x,S} > r_{x,T}$, or $r_{x,S} < r_{x,T}$. In the example of figure 4 we have $l_{x,S} > l_{x,T}$ for $x = 2, 3, 4, 5, 6$, hence $L_+ = 5$. Similarly $L_- = 5$, $R_+ = 5$, $R_- = 5$. In fact, for such trees all these values are equal to $n - 1$. We have:

Fact 4 $d(S,T) \geq \max(L_+, R_+) + \max(L_-, R_-)$.
Proof. As already noted, a right rotation causes the increase of one left and one right interval extreme, and a left rotation causes the decrease of one left and one right interval extreme. Then at least $max(L_-, R_-)$ right rotations, plus at least $max(L_+, R_+)$ left rotations, must be executed for the intervals of all vertices in S and T to become equal. *Q.E.D.*

Applying Fact 4 to the trees of figure 4 we have $d(S,T) \geq 5 + 5 = 10 = n - 1$. The same bound is found applying Corollary 1 (in fact $F_S = S$, $F_T = T$), or Lemma 2 (in fact $I(S,T) = n - 1$), therefore the new fact is not useful without a further observation. For the vertices $y = 2, 3, 4, 5, 6$ of S contributing to L_+, the values $l_{y,S}$ can be decreased only by a left rotation rot(x,y). However all these vertices, except for the last one 6, have the parent x at their right, so their left rotations require that a previous rotation be performed to create a parent at the left: in fact only a previous right rotation rot(x,y), or a previous left rotation rot(z,x), where z is the grandparent of y, allows a successive right rotation rot(z,y). Note now that none of such previous rotations influence the values of L_+ or R_-, then $L_+ - 1$ rotations must be performed in addition to the $\geq L_+ + R_-$ rotations needed bring the values of L_+ and R_- to zero. Then we have $d(S,T) \geq 4+5+5 = 14 = (n-1)/2 - 1 + 2(n-1)/2 = 3n/2 - 5/2$. For a tree of same shape with n even, such a bound immediately becomes $3n/2 - 2$. It is also interesting to note that this lower bound is sharp, because S is transformed into T by $3n/2 - 5/2$ rotations applying the algorithm of [6].

Let us now generalize the above observation. Let Λ be the set of vertices y with $l_{y,S} > l_{y,T}$ (i.e., they contribute to L_+) and the parent x is at the right of y. And let $\tilde{\Lambda}$ be the subset of vertices $y \in \Lambda$ such that $r_{y,S} \geq r_{y,T}$ (i.e., no vertex in $\tilde{\Lambda}$ contributes to R_-). In figure 4 we have $\Lambda = \{2, 3, 4, 5\}$ and $\tilde{\Lambda} = \Lambda$. Similarly let Γ be the set of vertices y with $r_{y,S} < r_{y,T}$ (i.e., they contribute to R_-) and the parent x is at the left of y. And let $\tilde{\Gamma}$ be the subset of vertices $y \in \Gamma$ such that $l_{y,S} \leq l_{y,T}$ (i.e., no vertex in $\tilde{\Gamma}$ contributes to L_+). In figure 4 we have $\Gamma = \{10, 9, 8, 7, 6\}$, and $\tilde{\Gamma} = \{10, 9, 8, 7\}$. By an immediate extension of the reasoning above we have:

Lemma 3 $d(S,T) \geq L_+ + R_- + \max(|\tilde{\Lambda}|, |\tilde{\Gamma}|)$.

For the trees of figure 4 both parameters of the function *max* have value 4, and we have the lower bound 14 already found.

Let us now sharpen the bound of Lemma 2. From Fact 1 we know that two consecutive vertices x, x+1 lie on the same path from the root to a leaf. Let $\pi(x, x+1)$ be the portion of that path between x and x+1 (we may have $x \downarrow x+1$ or $x+1 \downarrow x$), and let $\delta(x, x+1)$ be the length (number of edges) of $\pi(x, x+1)$. Fact 3 indicates that if x and x+1 form an inversion they must participate into the same rotation, therefore x and x+1 must become adjacent at a certain stage of the transformation. That is, if $\delta(x, x+1) > 1$, at least $\delta(x, x+1) - 1$ rotations must take place in S along $\pi(x, x+1)$ to reduce the length of this path to 1. A symmetrical condition holds after the rotation between x and x+1 has been executed, if the two vertices are not adjacent in T. If some of these rotations along the two paths in S and T must be made between vertices not forming

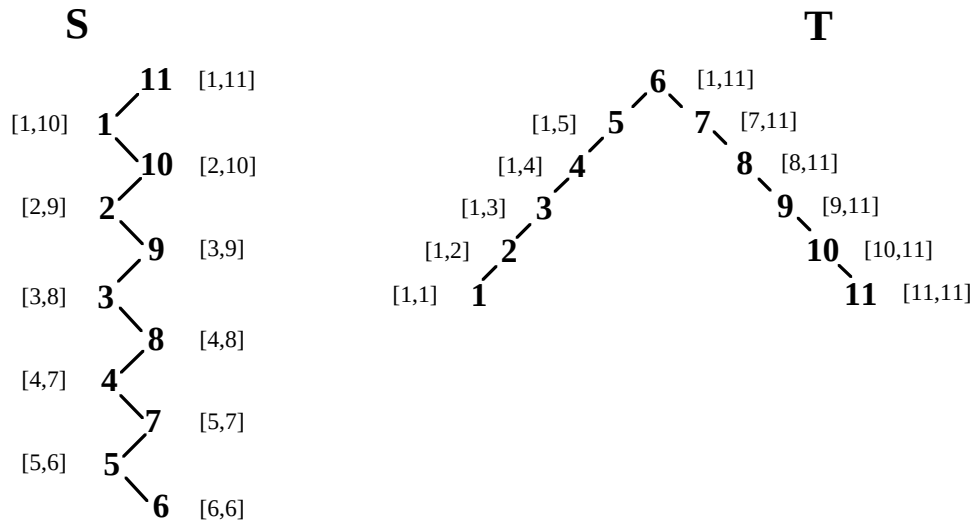

Figure 4: Two trees and the intervals of their vertices. For trees with this shape we have $d(S,T) = 3n/2-\mathrm{O}(1)$.

an inversion, their number must be added to the lower bound *I(S,T)* of Lemma 2. We must then count the minimum number of these extra rotations needed.

For the trees of figure 4 this count is easy. All pairs $x, x{+}1$ with $1 \leq x \leq n{-}1$ form an inversion, and all the corresponding paths in T have length 1, so we must examine only the paths in S. Out of these, $\pi(5,6)$ and $\pi(6,7)$ are not considered because the first has length 1, and the second has length 2 but can be shortened by 1 with a rotation between the vertices 5, 6 that form an inversion. All the other paths $\pi(x, x{+}1)$, with x equal 1 to 4, and 7 to 10, have length 2 and do not include pairs of inverted vertices, so in principle they contribute to increasing the lower bound. Note however that all such paths are pairwise overlapping, so only one half of them must be considered, because cutting one edge to one of them implies cutting the same edge to the overlapping path. This argument will be more formally proved in the next lemma. For the moment note that the paths $\pi(1,2)$, $\pi(2,3)$, $\pi(3,4)$, and $\pi(4,5)$ have no common edges and require one extra rotation each to be reduced to length 1, so at least four rotations must be added to the lower bound 10 of Lemma 2. The total lower bound then becomes 10+4=14 (or in general $n-1+(n-1)/2-1 = 3n/2-5/2$) as already found with Lemma 3. In general we have:

Fact 5 *Let the vertices x, $x{+}1$ form an inversion, and let σ_x be the number of pairs of vertices in $\pi(x, x{+}1)$ forming an inversion (including pair x, $x{+}1$). To bring x adjacent to $x{+}1$ at least $\tau_x = \delta(x, x{+}1){-}\sigma_x$ rotations between non inversion vertices are necessary along $\pi(x, x{+}1)$.*
Proof. Let p_x denote the path between x and $x{+}1$ at various steps of the transformation. And let d_x, s_x, t_x respectively denote the values of $\delta(x, x{+}1)$, σ_x, τ_x at the same steps. If during the transformation no vertex external to p_x enters this path the fact is immediately proved. Note now that for any vertex y external to p_x, vertex $y-1$, or $y+1$, but not both may be present in p_x, due to Fact 1. Then if y enters p_x at most one new pair of inversion vertices may appear in the path. That is d_x is increased by 1 and s_x is increased at most by 1, hence t_x is not decreased and the fact follows. *Q.E.D.*

As already noted, the rotations to bring x adjacent to $x{+}1$ may bring closer other pairs of inversion vertices whose paths overlap with $\pi(x, x{+}1)$. Then we have to individuate a set of inversion pairs whose paths do not share any edge. Formally let a *clean* subset of vertices $X =$

$\{x_1, ..., x_k\}$ be such that $x_i, x_i + 1$ is an inversion for $1 \leq i \leq k$; and $\pi(x_i, x_i{+}1)$, $\pi(x_j, x_j{+}1)$ have no common edges for $i \neq j$. And let $T_X = \sum_{i=1}^{k} \tau_{x_i}$, with τ_{x_i} defined as in Fact 5. We have immediately:

Lemma 4 *If X is a clean subset, then $d(S,T) \geq I(S,T) + T_X$.*

Ideally we are interested in a clean subset X whose T_X has maximal value. Since we are unable to give a polynomial time algorithm to find such a subset in the general case, we turn to using the best clean subset we can get heuristically. For the example of figure 4, the clean subset $X = \{1, 2, 3, 4\}$ already found has in fact maximal value $T_X = 4$ (or in general $T_X = (n-1)/2-1$), since the corresponding lower bound is sharp. Let us now examine a more interesting example.

For the trees of figure 3 we have $n = 2^4 + 2^3 = 24$ (or in general $n = 2^k + 2^{k-1}$). A clean subset X is given by the union of the two subsets $X_1 = \{5, 11, 17, 23\}$ in S and $X_2 = \{1, 7, 13, 19\}$ in T, whose paths are indicated with dots in the figure. Note that the only inversion pairs appearing in these paths are constituted by the extreme vertices and we have: $\tau_5 = 1$, $\tau_{11} = 2$, $\tau_{17} = 1$, $\tau_{23} = 3$ in X_1; $\tau_1 = 3$, $\tau_7 = 1$, $\tau_{13} = 2$, $\tau_{19} = 1$ in X_2. In total we have $T_X = T_{X_1} + T_{X_2} = 14$ (or in general $T_X = 2^k - 2$). By Lemma 4 we then have: $d(S,T) \geq I(S,T) + T_X = 23 + 14 = 37$, or in general: $d(S,T) \geq 2^k + 2^{k-1} + 2^k - 2 = 5n/3 - 3$.

With some ingenuity it can be shown that this is also an upper bound to $d(S,T)$, i.e. the lower bound of Lemma 4 is sharp for trees of this shape.

4 Concluding remarks.

The initial goal of this work was finding two trees S, T with a "regular" structure, leading to establish a lower bound of $2n - 6$ (or at least $2n - \mathrm{O}(1)$) to the rotation distance between S and T for all values of $n > 11$, thus proving the conjecture of [9]. An even more ambitious goal was casting some light on an optimal algorithm for computing the rotation distance. We did not secceed in either direction, so these problems remain substantially open.

A complete enumaration conducted for $11 \leq n \leq 20$ shows that there is an exceedingly large number of pairs of trees requiring $2n - 6$ rotations for their transformation [3], but none of the pairs that we could examine thus far exhibits a structure apt to derive the matching lower bound. This could be the reason why proving this bound appears to be so difficult. In any case we believe that further work has to be done along the line of this paper.

Finally, it is worth noting that an interesting combinatorial study on the rotation distance has recently appeared in the Web, as a manuscript yet unpublished in paper [2]. In particular it exhibits trees with a lower bound of $2n - O(\sqrt{n})$, in addition to several other examples. The techniques and results presented in [2] are quite complex and completely different from ours, so the two studies may be regarded as being complementary.

References

[1] K. Culik and D. Wood, A note on some tree similarity measures, *Information Processing Letters* 15 (1982) 39-42.

[2] P. Dehornoy, On the rotation distance between binary trees, (2009) http://arxiv.org/abs/0901.2557

[3] E. Del Tessandoro, The upper bound on the rotation distance of binary tree: an experimental study, *Dipartimento di Informatica, University of Pisa*. Report on a Project for B.S. Degree in Informatics (2009).

[4] J. Lucas, Untangling binary trees via rotations, *The Computer Journal* 47 (2) (2004) 259-269.

[5] F. Luccio and L. Pagli, On the upper bound on the rotation distance of binary trees, *Information Processing Letters* 31 (2) (1989) 57-60.

[6] E. Mäkinen, On the rotation distance of binary trees, *Information Processing Letters* 26 (5) (1988) 271-272.

[7] J. Pallo, An efficient upper bound of the rotation distance of binary trees, *Information Processing Letters* 73 (3-4) (2000) 87-92.

[8] R. Rogers, On finding shortest paths in the rotation graph of binary trees, in: *Proc. Southeastern Internat. Conf. on Combinatorics, Graph Theory, and Computing*, Vol. 137 (1999) 77-95.

[9] D.D. Sleator, R.E. Tarjan, W.R. Thurston,Rotation distance, triangulations, and hyperbolic geometry, *J. Amer. Math. Soc.* 1 (3) (1988) 647-681.