On the diameter of the rotation graph of binary coupling trees

V. Fack, S. Lievens and J. Van der Jeugt[1]
Department of Applied Mathematics and Computer Science,
University of Ghent, Krijgslaan 281-S9, B-9000 Gent, Belgium

Abstract

A binary coupling tree on $n+1$ leaves is a binary tree in which the leaves have distinct labels. The rotation graph G_n is defined as the graph of all binary coupling trees on $n+1$ leaves, with edges connecting trees that can be transformed into each other by a single rotation. In this paper we study distance properties of the graph G_n. Exact results for the diameter of G_n for values up to $n = 10$ are obtained. For larger values of n we prove upper and lower bounds for the diameter, which yield the result that the diameter of G_n grows like $n \lg(n)$.

Corresponding author: J. Van der Jeugt, Department of Applied Mathematics and Computer Science, University of Ghent, Krijgslaan 281-S9, B-9000 Gent, Belgium.

Tel. ++ 32 9 2644812; Fax ++ 32 9 2644995; E-mail Joris.VanderJeugt@rug.ac.be.

[1]Research Associate of the Fund for Scientific Research – Flanders (Belgium)

1 Introduction

In computer science and discrete mathematics, one often faces the problem of transforming one configuration into another by specified rules. The question arises of how many steps might be needed, in the worst case. This is modeled graph-theoretically by letting the configurations be the vertices of a graph whose edges correspond to the allowed steps. The question is then to determine the diameter of this graph. In this paper, we consider a family of these problems where the configurations are binary trees with the same number of leaves.

Binary trees are of fundamental importance in graph theory and in various branches of applied mathematics and computer science. The trees that occur most often are the binary plane trees, being associated with binary search trees. (In a binary tree, every node has zero or two children; in a plane tree, the children of a node have a fixed left-to-right order.) The number of binary rooted plane trees with $n + 1$ leaves is the nth Catalan number. On the set of binary rooted plane trees with a fixed number of leaves, one can define a "rotation" that transforms one tree into another. A fundamental question is to find the number of rotations needed to transform one such tree into a second one. Often this problem is formulated as a graph distance problem: the graph is defined on the set of binary rooted plane trees with $n + 1$ leaves, and adjacency is determined by the rotation operation. It has been shown that the diameter of this graph is bounded by $2n - 6$; computing the actual distance between two given trees remains a difficult problem [18, 13, 12, 16].

Inspired by this problem, and motivated by two applications, we consider in this paper a similar problem. The trees appearing here are ordinary (i.e. not plane) binary rooted trees with $n+1$ labeled leaves. The number of such trees is given by $(2n-1)!! = 1\cdot 3\cdot \ldots \cdot (2n-1)$. We consider a graph G_n defined on the set of such trees, and also define adjacency by a "rotation" operation that transforms one tree into another. This operation models transformations between objects modeled by the trees. In various applications (e.g. generalized recoupling coefficients in quantum theory of angular momentum [5], computation of a similarity measure between dendrograms [20]) the question of how many operations are needed to turn one object into another is of interest. Thus we study the diameter of this graph.

The structure of the paper is as follows. Section 2 defines the trees we are dealing with (referred to as binary coupling trees) and the rotation graph G_n, and describes some basic properties of the graph G_n. In Section 3 exact results for some distance properties (such as distance degree sequence and diameter) are given for small values of n ($n \leq 10$). The size of G_n is growing exponentially in n, so for large values of n we look for theoretical bounds for the diameter of G_n. In Section 4 we obtain an explicit upper bound by constructing a path between two arbitrary binary coupling trees and by showing that its length is necessarily bounded by $n\lg(n) + O(n)$. Section 5 shows how an $\Omega(n\lg(n))$ lower bound for the diameter can be obtained from an upper bound for the number of trees within a certain distance of any given tree, for which the technique of short encodings introduced by Sleator *et al* in [17] can be used. We conclude that the diameter of G_n is $\Theta(n\lg(n))$. In particular, we will prove the following theorem:

Theorem 1 *For $n \geq 1$, the diameter* $\mathrm{diam}(G_n)$ *of G_n satisfies*

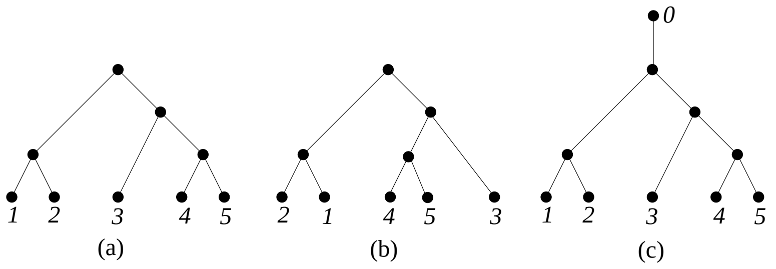
$$\frac{1}{4}n\lg(\frac{n}{e}) < \mathrm{diam}(G_n) < n\lceil \lg n \rceil + n - 2\lceil \lg n \rceil + 1.$$

2 Binary coupling trees and the graph G_n

We define a *binary coupling tree* as a binary tree in which the leaves (i.e. nodes with no children) are given distinct labels. Without loss of generality, we can assume that these labels are the integer numbers between 1 and $n+1$ if the binary coupling tree has $n+1$ leaves. For fixed $n \geq 1$, we denote the set of all binary coupling trees with $n+1$ leaves, or equivalently with n non-leaf nodes, as $\mathcal{T}_n$. Figure 1(a) and (b) give two drawings of the same binary coupling tree. Note that one can place the children of a node in a binary coupling tree in any order; i.e. binary coupling trees are *not* plane trees. Sometimes, it will be convenient to attach an extra leaf with label 0 to the root and regard the binary coupling tree as an unrooted tree in which every node has degree 1 or 3; this is shown in Figure 1(c). We call these *extended binary coupling trees* and use $\tilde{\mathcal{T}}_n$ to denote the set of extended binary coupling trees with $n+2$ leaves.

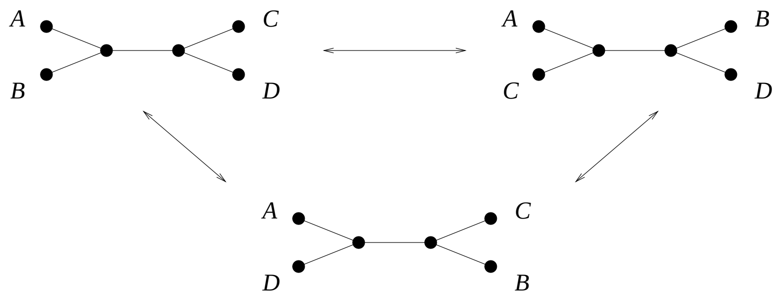
An edge joining non-leaf nodes is an *internal edge.* In an extended binary coupling

Figure 1: Binary coupling trees



tree, the two nodes of an internal edge are adjacent to four other nodes. There are three pairings of four elements. A *rotation* allows these four nodes to be paired in one of the two other ways. There are thus two rotations around an internal edge. Figure 2 gives an illustration; here each of A, B, C, D stands for a leaf or an arbitrary subtree. Note that a rotation is invertible; if T_2 is obtained by performing a rotation on T_1, then T_1 can be obtained by performing a rotation on T_2. This is also indicated in Figure 2. In the literature, other names for rotations appear: *flops* [6], *nearest neighbour interchanges* [4] and *crossovers* [15]. Note that when *plane* binary trees are studied there is only one rotation available at each internal edge.

Figure 2: Rotations on binary coupling trees



For fixed $n \geq 1$, we build the *rotation graph* G_n as follows: each vertex of G_n represents an element from $\mathcal{T}_n$. Two vertices are adjacent if and only if the two binary coupling trees they represent are related through a single rotation. Some simple properties of G_n were proved in [15]; see also [5]. We summarize them here:

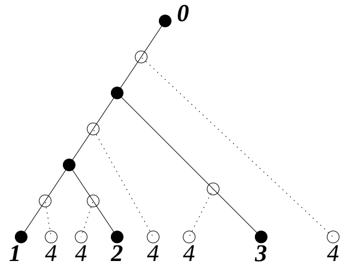
- $|V(G_n)| = |\mathcal{T}_n| = (2n-1)!! = 1 \cdot 3 \cdot \ldots \cdot (2n-1)$,

- G_n is regular of degree $2(n-1)$,
- G_n is connected.

To see that $|\mathcal{T}_n| = (2n-1)!!$, consider an element T of $\tilde{\mathcal{T}}_{n-1}$. The tree T has $2n-1$ edges, so there are $2n-1$ different ways of subdividing an existing edge and attaching an extra edge with leaf label $n+1$ to the new vertex (see Figure 3). Furthermore, each element of $\tilde{\mathcal{T}}_n$ arises exactly once in this way. Thus we have

$$|\tilde{\mathcal{T}}_n| = |\mathcal{T}_n| = (2n-1)|\tilde{\mathcal{T}}_{n-1}| = (2n-1)!!.$$

Figure 3: Five ways of attaching an extra leaf label 4 to an element of $\tilde{\mathcal{T}}_2$



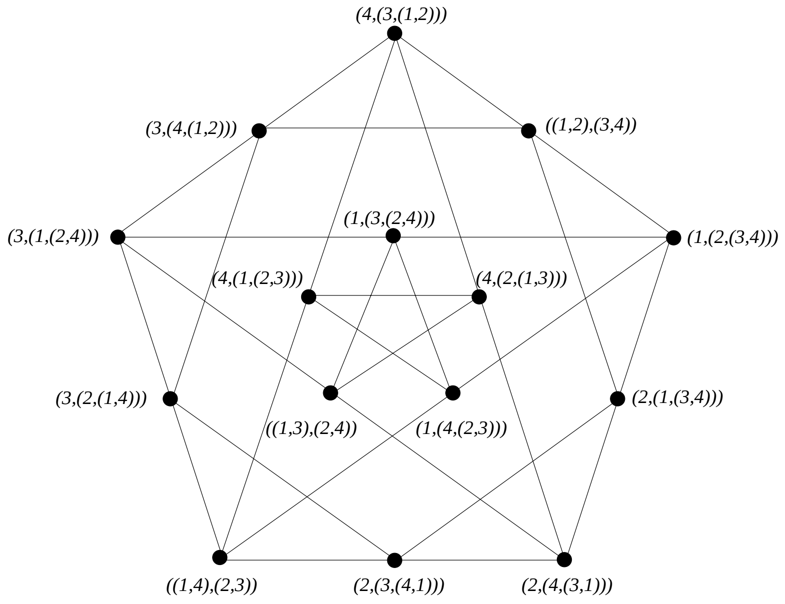
Example 2 As can be seen in Figure 4, the graph G_3 has $1\cdot 3\cdot 5 = 15$ vertices, while every vertex has four neighbours. In Figure 4, every vertex is labeled with a *bracket notation* of the binary coupling tree it represents. A bracket notation of a binary coupling tree gives the way in which the labeled leaves are coupled to form the binary coupling tree. Possible bracket notations of the binary coupling tree in Figure 1(a) are:

$$((1,2),(3,(4,5))) \qquad \text{or} \qquad ((2,1),((4,5),3)).$$

∎

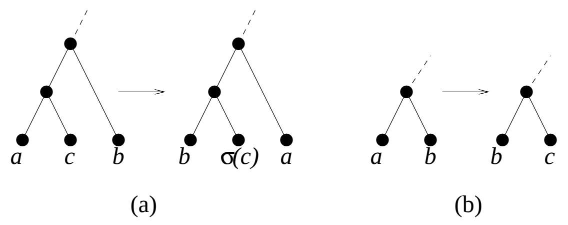
Let σ be an element of S_{n+2}, the group of all permutations on $n+2$ elements; σ acts on $T \in \tilde{\mathcal{T}}_n$ (and on G_n) by permutation of the $n+2$ leaf labels. It is clear that if T_1 and T_2 (viewed as elements of $\tilde{\mathcal{T}}_n$) are adjacent in G_n, then $\sigma(T_1)$ and $\sigma(T_2)$ are also adjacent in G_n. Thus $\sigma(G_n)$ is isomorphic to G_n. Furthermore, for $n \geq 3$ no element of S_{n+2} except

Figure 4: The rotation graph G_3



the identity permutation fixes G_n completely. Indeed, if σ has a cycle (ab) of length 2, then all trees of the form indicated in Figure 5(a) are not fixed under σ. If σ has no cycle of length 2 and $\sigma \neq \text{id}$, then it must have a cycle $(abc\ldots)$ of length > 2. In this case, all trees of the form indicated in Figure 5(b) are not fixed under σ.

Figure 5: Trees that are not fixed under σ



Thus, we can conclude that for $n \geq 3$, the automorphism group of G_n contains S_{n+2}. For $n \in \{3, 4, 5, 6\}$, equality holds; we have verified this using the `nauty` program [14]. For larger values of n, the question of whether equality holds remains open.

3 Distance in G_n

In this paper, we are primarily concerned with computing or estimating the diameter of G_n (the *diameter* $\operatorname{diam}(G)$ of a graph G is the maximum over $v, v' \in V(G)$ of the distance $d(v, v')$).

The diameter and many other concepts related to distance (eccentricity, radius, center, periphery, ... [2]) follow easily if we know the *distance degree sequence* for every vertex of G_n. The distance degree sequence for a vertex v of G_n is the sequence

$$dds(v) = (d_0(v), d_1(v), d_2(v), \ldots),$$

where $d_i(v)$ is the number of vertices at distance i from v.

It is obvious that many vertices of G_n give rise to the same distance degree sequence. When two binary coupling trees differ only by a permutation of their labels, we say they have the same *type*. Clearly, such trees have the same distance degree sequence. As indicated in [5] and in Figure 6(a), there are two different types of binary coupling trees on 4 leaves, yet the distance degree sequence of these two types is identical. This can be understood by considering the corresponding elements from $\tilde{\mathcal{T}}_n$; indeed, these elements differ only by a permutation of their labels, see Figure 6(b). The *skeleton* of an extended binary coupling tree is the tree obtained by deleting all leaves from the extended binary coupling tree, see Figure 6(c). In other contexts, the skeleton of an extended binary coupling tree has been called its 'derived tree'. Two extended binary coupling trees differ only by a permutation of the leaf labels if and only if their skeletons are isomorphic. The skeletons of elements of $\tilde{\mathcal{T}}_n$ are precisely the isomorphism classes of trees with n nodes in which every node has degree at most 3.

To determine the diameter of G_n for some small fixed n, it is sufficient to calculate the distance degree sequence for all skeletons with n nodes. Table 1 lists the number of types and skeletons for values of n up to 10. The sequence giving the number of types is sequence A001190 of [19]; it is also known as the Wedderburn-Etherington sequence. The number of skeletons is sequence A000672 of [19]. The number of skeletons is (much) smaller than the number of types, yielding a substantial decrease in the required computation time. This reduction technique was used by Jarvis *et al* [8]. Distance degree sequences

Figure 6: (a) The two types in $\mathcal{T}_3$, (b) their corresponding extended binary coupling trees and (c) the corresponding skeletons

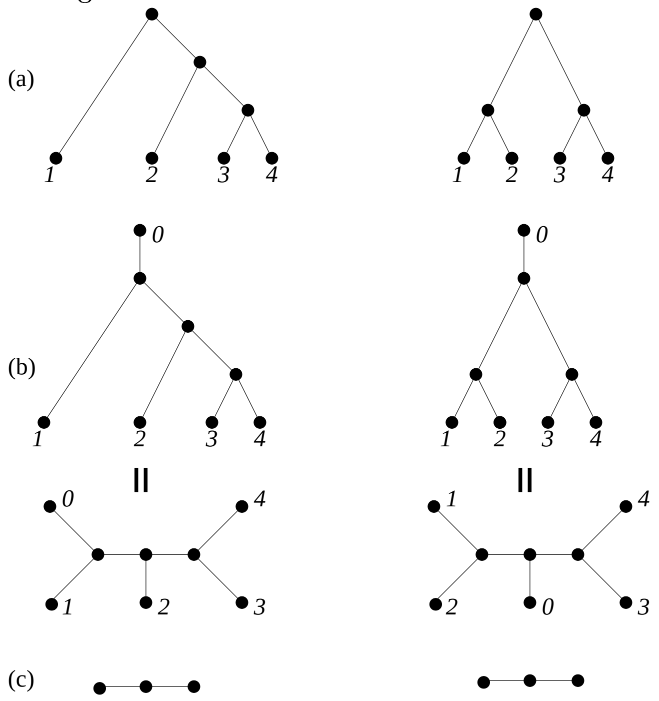


Table 1: Number of types and skeletons for $n \leq 10$

n	2	3	4	5	6	7	8	9	10
types	1	2	3	6	11	23	46	98	207
skeletons	1	1	2	2	4	6	11	18	37

up to $n = 7$ are given in [5]; the complete results up to $n = 10$ can be found at URL `http://allserv.rug.ac.be/~jvdjeugt/BCT`. The diameter of G_n for $n \leq 10$ is shown in Table 2.

Table 2: Diameter of G_n for $n \leq 10$

n	2	3	4	5	6	7	8	9	10
diam(G_n)	1	3	5	7	10	12	15	18	21

4 An upper bound for the diameter of G_n

For $T_1, T_2 \in \mathcal{T}_n$, we will construct a path between the corresponding vertices in G_n. Robinson [15], Culik and Wood [3], and Li *et al* [11] used the same technique to obtain $O(n^2)$, $4n\lg(n) + O(n)$, $n\lg(n) + O(n)$ upper bounds for the diameter of G_n respectively. Here, and in the rest of this paper, lg denotes the logarithm in base 2. We will follow the lines indicated in [11] to obtain an *explicit* upper bound of the form $n\lg(n) + O(n)$ for the diameter of G_n; in particular, we will make the "$O(n)$" part explicit by performing a more careful calculation.

Our approach to obtain an upper bound is a slight modification of the standard approach to bounding the diameter by showing that all vertices are within a fixed distance of a single vertex. Here, we show that all vertices are within a fixed distance of a special set of vertices, and we give an upper bound for the diameter of this set. The reason for the variation here is the labeling of the leaves: the special set consists of different labelings of a single isomorphism class.

The *level of a node in a tree* is defined recursively as follows [10, Section 2.3]: the level of the root is zero and the level of any other node is one more than the level of its parent.

The *depth of a tree* T, denoted as $\mathrm{depth}(T)$, is the maximum level of any of its nodes. For a rooted binary tree T with $n+1$ leaves, it is well known that

$$\lceil \lg(n+1) \rceil \leq \mathrm{depth}(T) \leq n. \tag{1}$$

An element $S \in \mathcal{T}_n$ is a *spine* if and only if $\mathrm{depth}(S) = n$. Spines exist for every $n \geq 1$; indeed, there are $\frac{(n+1)!}{2}$ spines in $\mathcal{T}_n$.

The path between T_1 and T_2 is constructed in three steps:

1. transform T_1 into a spine S_1,
2. transform T_2 into a spine S_2 and,
3. transform the spine S_1 into S_2 (or vice versa).

In this section, we will determine an explicit upper bound for the number of rotations needed in each step, yielding an explicit upper bound for the diameter of G_n.

Let T be a binary coupling tree that is not a spine. Choose a leaf x of T that has maximum level. Since T is not a spine, there is an internal edge of T that is not on the path from the root node of T to x, but that has a node in common with an edge on this path. Performing the appropriate rotation around this internal edge will increase the depth of T by one. Hence, one can transform an arbitrary element T of $\mathcal{T}_n$ into a spine using $n - \mathrm{depth}(T)$ rotations.

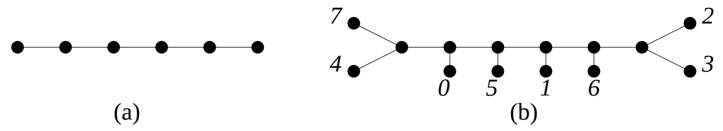
Thus, given the bound in (1), one can transform any binary coupling tree on $n+1$ leaves into a spine using at most

$$n - \lceil \lg(n+1) \rceil \tag{2}$$

rotations.

The construction of a path between two arbitrary spines from $\mathcal{T}_n$ is easier to understand when working with extended binary coupling trees, i.e. elements of $\tilde{\mathcal{T}}_n$. We say that an element S from $\tilde{\mathcal{T}}_n$ is an *extended spine* if and only if its skeleton is a path. Figure 7(a) is a drawing of a path on six nodes, while Figure 7(b) is a drawing of an extended spine of $\tilde{\mathcal{T}}_6$. Note that an extended spine corresponds to a spine if and only if the label 0 appears on a leaf at the end of the path.

Figure 7: (a) A path on six nodes and (b) an extended spine of $\tilde{\mathcal{T}}_6$



Rotations on extended binary coupling trees are rotations of the corresponding binary coupling trees. Thus the maximum distance between extended spines in $\tilde{\mathcal{T}}_n$ is an upper bound for the maximum distance between spines in $\mathcal{T}_n$ (it may be larger since the set of extended spines is larger). By symmetry (relabeling of leaves), it suffices to bound the distance of all extended spines from a fixed extended spine.

A rotation that transforms one extended spine into another performs (except at the ends) an adjacent transposition on the permutation recording the leaves. Thus $\Theta(n^2)$ rotations may be needed to transform one extended spine into another using extended spines only. This corresponds to simulating a bubble sort [9, Section 5.2.2] on the extended spines and leads to an $O(n^2)$ upper bound for the diameter of G_n. In order to reduce the bound to $O(n \lg n)$, it is necessary to use vertices outside the set of extended spines. The faster method simulates the merge sort algorithm [9, Section 5.2.4] on the set of extended spines.

An extended spine of $\tilde{\mathcal{T}}_n$ has four *end leaves*, i.e. leaves whose neighbour is an endpoint of the skeleton; in Figure 7(b), these are the leaves with labels 7, 4, 2, and 3. Let S be an extended spine, and let x be an end leaf. We say that S is *increasing* (resp. *decreasing*) *with respect to* x if and only if for all other leaves x_1 and x_2 the following property holds:

$$d(x_1, x) < d(x_2, x) \Rightarrow x_1 < x_2 \text{ (resp. } x_1 > x_2).$$

Herein, x_i denotes both the leaf x_i and the label of this leaf. If the leaf label of x is known, then there is exactly one extended spine in $\tilde{\mathcal{T}}_n$ that is increasing with respect to x; we will use this extended spine as the fixed spine mentioned before.

Let S be an extended spine of $\tilde{\mathcal{T}}_n$, and let x be an end leaf. We say that $S \in \tilde{\mathcal{T}}_n$ is *concave with respect to* x (resp. *convex with respect to* x) if and only if for all other leaves

x_1 and x_2 the following property holds:

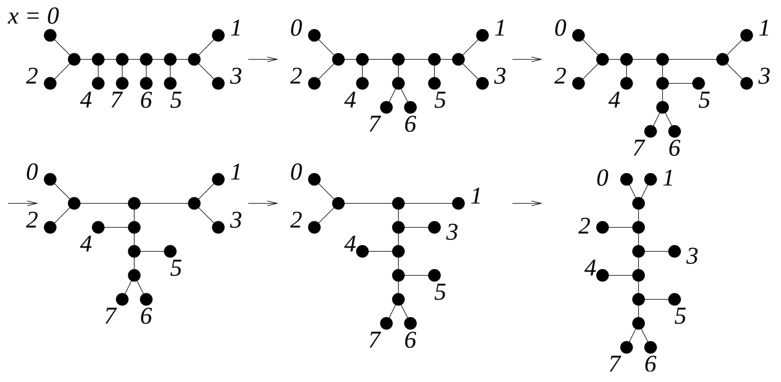
$$d(x_1, x) < d(x_2, x) \leq \left\lceil \frac{n}{2} \right\rceil + 1 \Rightarrow x_1 < x_2 \text{ (resp. } x_1 > x_2\text{)}$$

and

$$\left\lceil \frac{n}{2} \right\rceil + 2 \leq d(x_1, x) < d(x_2, x) \Rightarrow x_1 > x_2 \text{ (resp. } x_1 < x_2\text{)}.$$

If $S \in \tilde{\mathcal{T}}_n$ is an extended spine that is concave (resp. convex) with respect to x, then we can transform S into an increasing (resp. decreasing) extended spine, again with respect to x, using at most $n-1$ rotations. This procedure, illustrated in Figure 8, is quite analogous to the *merge* step in the merge sort algorithm, where two sorted sequences are combined to form a single sorted sequence; it uses induction on n. When $n = 2$, at most

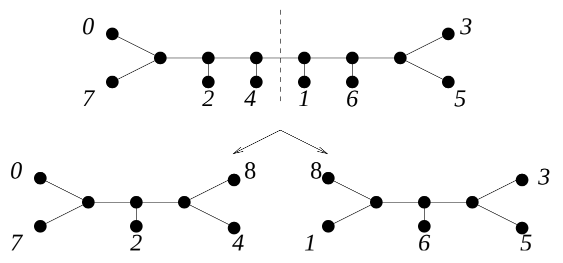
Figure 8: Merging an extended spine



one rotation is needed (see Figure 2). For $n \geq 3$, consider the two leaves with the largest labels (excluding x) in an extended spine concave with respect to x. Since the neighbours of these two leaves are adjacent, we can perform a rotation that gives the two leaves a common neighbour. We then delete these two leaves and give their neighbour (which is now a leaf) the label of the smaller one. In this way, we have obtained an extended spine of $\tilde{\mathcal{T}}_{n-1}$ that is concave with respect to x. By induction, at most $n-2$ rotations are needed to transform this extended spine into an increasing one. Since the leaf with the largest label appears at the end of the extended spine, replacing this leaf with the two original leaves will produce an extended spine of $\tilde{\mathcal{T}}_n$ increasing with respect to x.

Also the other ideas of the merge sort algorithm apply to our problem. Let $S \in \tilde{\mathcal{T}}_n$ be an extended spine that is to be transformed into an increasing or a decreasing one with respect to some leaf x. We make an imaginary cut on S and obtain two extended half-spines by placing an imaginary leaf on each end of the cut. The label we place on the imaginary leaves is the same for both halves and depends on whether we are transforming S into an increasing or a decreasing extended spine. In the former case the value of the label of the imaginary leaves is greater than any other label of the extended spine, while in the latter it is smaller. In this way, we get two extended half-spines: one in $\tilde{\mathcal{T}}_{\lceil \frac{n}{2} \rceil}$, containing the leaf x, and one in $\tilde{\mathcal{T}}_{\lfloor \frac{n}{2} \rfloor}$. (To really match the definition, we would have to do an order-preserving relabeling.) Figure 9 illustrates how we place the imaginary cut when the extended spine is to be transformed into an increasing one with respect to the leaf 0.

Figure 9: Placing an imaginary cut on an extended spine



If we are transforming S into an increasing (resp. decreasing) extended spine with respect to the leaf x, we *recursively* transform the extended half-spine containing x into an increasing (resp. decreasing) one with respect to the original leaf x, while we recursively transform the other extended spine into a decreasing (resp. increasing) one with respect to the imaginary leaf. Once we have done this, we can merge the two extended half-spines together.

Each time the sorted subspines double in length, at most $n-1$ rotations are performed. We thus expect that approximately $n \lg n$ rotations are needed to sort an extended spine.

Let $f(n)$ (resp. $g(n)$) denote the maximum number of rotations needed to transform an arbitrary extended spine $S \in \tilde{\mathcal{T}}_n$ into an increasing (resp. decreasing) one with respect to some fixed leaf x. By symmetry, it is clear that $f(n) = g(n)$ for all values of $n \geq 1$. Since $|\tilde{\mathcal{T}}_1| = 1$, $f(1)$ equals 0. Hence f satisfies

$$f(n) \leq f\left(\left\lceil \frac{n}{2} \right\rceil\right) + f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n - 1, \text{ for } n > 1 \text{ and } f(1) = 0. \qquad (3)$$

Let $f_u(n)$ denote the function for which equality holds in (3), i.e.:

$$f_u(n) = f_u\left(\left\lceil \frac{n}{2} \right\rceil\right) + f_u\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n - 1, \text{ for } n > 1 \text{ and } f_u(1) = 0. \tag{4}$$

This is a well-known recurrence [7, Section 3.3] and its solution is given by:

$$f_u(n) = n\lceil \lg n \rceil - 2^{\lceil \lg n \rceil} + 1, \tag{5}$$

which is indeed approximately $n \lg(n)$ as expected. As already noted, $f(n)$ is an upper bound for the number of rotations needed to transform an arbitrary spine $S_1 \in \mathcal{T}_n$ into another arbitrary spine $S_2 \in \mathcal{T}_n$. We thus have the following theorem:

Theorem 3 *The diameter of G_n satisfies*

$$\operatorname{diam}(G_n) \leq n\lceil \lg(n) \rceil - 2^{\lceil \lg(n) \rceil} + 1 + 2\left(n - \lceil \lg(n+1) \rceil\right). \tag{6}$$

Proof: This follows immediately by combining formulas (5) and (2). □

5 A lower bound for the diameter of G_n

Upper bounds on the number of vertices within distance m of an arbitrary vertex in a graph G yield lower bounds on the diameter of G.

Such a bound is easily obtained by considering the following inequalities that hold for any vertex v of a graph G with maximal degree Δ:

$$d_0(v) = 1, \quad d_1(v) \leq \Delta, \quad d_i(v) \leq \Delta(\Delta - 1)^{i-1}, \text{ for } i > 1.$$

This gives the following inequality:

$$\frac{\Delta(\Delta - 1)^{\operatorname{diam}(G)} - 2}{\Delta - 2} \geq \sum_{i=0}^{\operatorname{diam}(G)} d_i(v) = |V(G)|. \tag{7}$$

This bound on the order of graphs with fixed maximum degree and diameter is known as the *Moore bound*. Graphs for which equality holds in (7) are *Moore graphs* and are extremely rare (see [1, 2] for further discussion). If we apply inequality (7) to the rotation graph G_n, we get a linear lower bound for the diameter of G_n, namely

$$\operatorname{diam}(G_n) \geq \frac{\ln(2n) - 1}{\ln(2n)} n. \tag{8}$$

Li *et al* [11] proved an $\Omega(n\lg(n))$ lower bound for $\operatorname{diam}(G_n)$ using the results of [17]. They sketched a way, using "flips" in plane triangulations and short encodings, to derive that the number of trees within distance m from any given tree is bounded by $3^n 2^{4m}$. We will show that the number of trees within distance m is bounded by

$$\frac{2^{n+4m}}{2n}.$$

Since this is smaller than $3^n 2^{4m}$, our lower bound will be better than the one found by Li *et al.* We will prove in particular that for $n > 1$, the following inequality holds:

$$\operatorname{diam}(G_n) > \frac{1}{4}\lg(n!) > \frac{1}{4}n\lg(\frac{n}{e}).$$

In [17] Sleator *et al* provide a tool for deriving an upper bound for the number of combinatorial objects within m transformations from a given object. They take advantage of the fact that often one can interchange the order of the transformations without affecting the final outcome. This does not imply that all lists of m transformations that reach a given object are reorderings of each other: it is also possible to reach the given object using different sets of transformations.

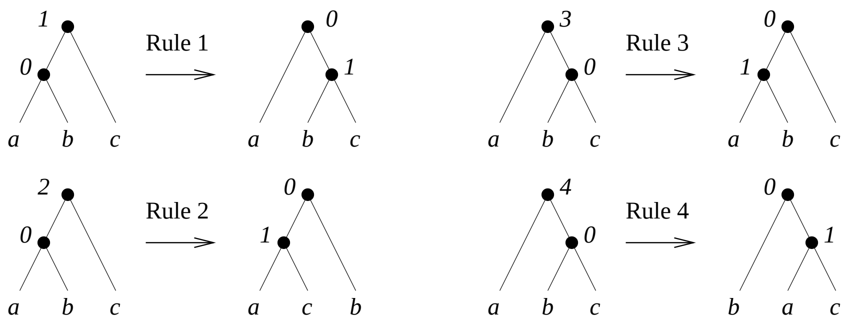
We will apply their technique of *short encodings* to paths in G_n. We will encode every path starting from a particular tree as a list of integers in $\{0,1,2,3,4\}$, and then we will bound the number of encodings for paths of length at most m by $2^{n+4m}/(2n)$.

As already noted, a binary coupling tree does not change if one exchanges the "left" and "right" child of any non-leaf node. This transformation is called an *exchange* [6] or a *twist* [17]. In the technique following from Sleator *et al* [17] the trees are ordered, so the twist transformation is also counted. Here however, we do *not* want to count twists. This problem can be overcome by working with ordered trees for which a "twist-rotation-twist"-transformation is counted as one transformation only. That is why two transformations will be added to the ordinary rotation transformation (see Figure 10).

Let T and T' be elements of $\mathcal{T}_n$ with $d(T,T') = m$. This means that there exists a sequence of m rotations that carries T into T'; this sequence is called a *derivation.* Note that there may be many derivations that carry T into T'.

In a derivation, we view the trees along the way as *ordered* trees, i.e. twists are not allowed. When regarding T as an ordered tree, we denote it as $\hat{T}$. We can *apply* one of the four *rules* (transformations) indicated in Figure 10 to $\hat{T}$ if and only if $\hat{T}$ contains a subtree identical to the tree on the left side of that rule (temporarily ignore the labels on the internal nodes). The result is $\hat{T}$ in which the left side of the rule is replaced by the right side, so the left (resp. right) side of the rule relates to the shape of the tree *before* (resp. *after*) the rule is applied. The "pure rotations" applied to these ordered trees correspond to Rules 1 and 3. The other rules correspond to a "twist-rotation-twist" transformation. For example, Rule 2 corresponds to a twist (exchange a and b), followed by a rotation, followed by another twist (exchange b with parent node of a and c). Operating on ordered trees, these four rules are necessary and sufficient to produce all possible rotations on binary coupling trees. The numbers of the nodes in the left sides of the rules are called *pre-position numbers*, while the numbers of the nodes in the right sides of the rules are called *post-position numbers*. Their use will soon become apparent.

Figure 10: The four rules that can be applied



It is convenient to think of applying a rule as destroying nodes and creating new ones. To keep track of this process, we assign distinct *names* to the non-leaf nodes in the trees produced during a derivation. An *action* is an application of a rule to particular nodes, so a derivation is a list of actions. The *required nodes* of an action are the nodes that are destroyed by that action. An action is *ready* if and only if the required nodes of that action exist.

In order to name each non-leaf node that appears in the trees produced by a derivation, we first number the actions of that derivation, beginning with 1. Each internal node of the initial tree $\hat{T}$ is named v_i, with $1 \leq i \leq n$, in some (arbitrary) order. Next, each new node gets a name of the form $v_{j,0}$ or $v_{j,1}$, where j is the number of the action that created these nodes and where 0 and 1 refer to the post-position numbers of the applied rule.

Figure 11: A derivation of length 4

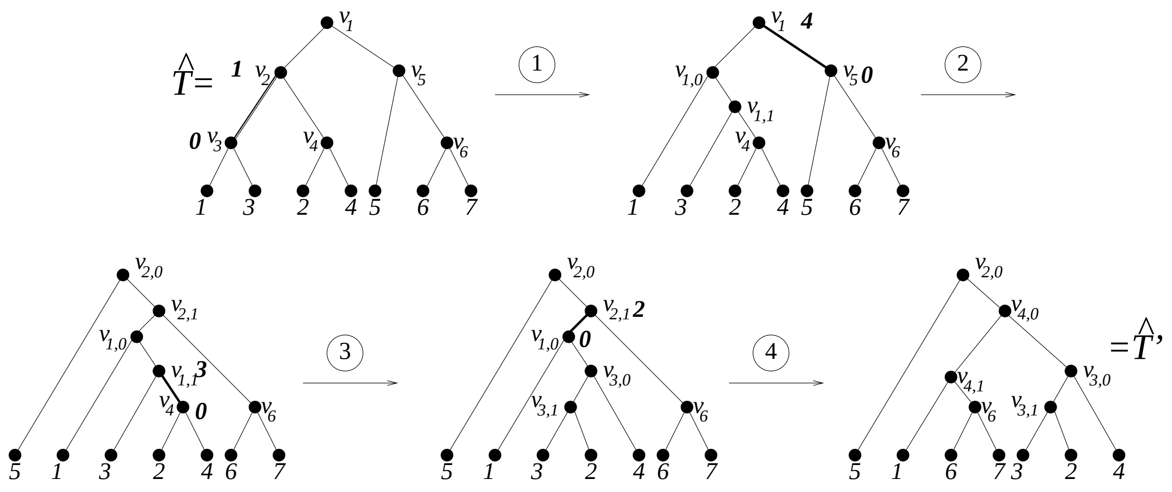


Table 3: Required nodes for each action of the derivation in Figure 11

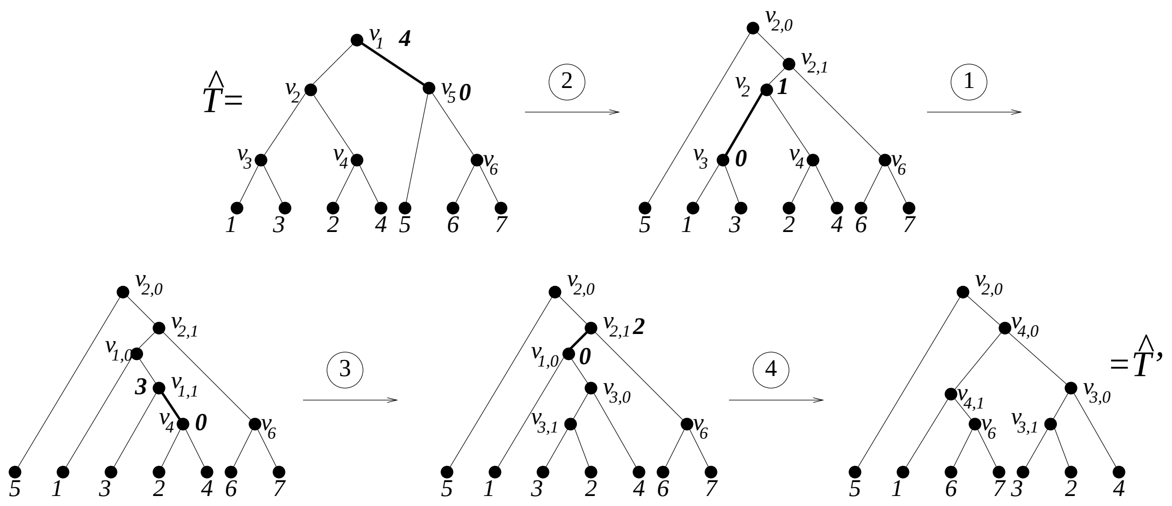
action	required nodes
1	v_2, v_3
2	v_1, v_5
3	v_4, $v_{1,1}$
4	$v_{1,0}$, $v_{2,1}$

In order to build an encoding for the derivation D with initial tree $\hat{T}$, we first (a) number the actions of D, (b) give each internal node a name, and (c) determine the required nodes of each action. Furthermore, we associate with the name of each required node the pre-position number of the corresponding node in the rule applied to that required node. If no rule is ever applied to a node, then that node survives in $\hat{T}'$, and we associate 0 with the name of that node. These numbers are determined by which rule is applied, not the index of the action, so they lie in $\{0, 1, 2, 3, 4\}$.

In order to encode a derivation D, we first construct a canonical derivation D' that is a reordering of the actions of D and produces the same final outcome $\hat{T}'$. To select the next action for D' from the remaining unprocessed actions of D, at each step we choose from the actions that are ready the action that destroys the node with the smallest name in lexicographic order. This lexicographic order treats the initial single-coordinate names as being smaller than all names that are introduced later. Having done this until all actions are applied, the encoding of D now consists of the pre-position numbers associated with the internal nodes, *in the order introduced by the canonical derivation* D'.

Example 4 For the derivation in Figure 11, Tables 3 and 4 give the required nodes of each action and the association of the names with pre-position numbers. Looking at the initial tree, or equivalently at Table 3, we see that actions 1 and 2 are ready. We choose to do action 2 first, because action 2 destroys the node with the smallest name. Thus, nodes v_1 and v_5 are destroyed and nodes $v_{2,0}$ and $v_{2,1}$ are created. Next, only action 1 is ready so we do action 1, hereby destroying the nodes v_2 and v_3 and creating the nodes $v_{1,0}$ and $v_{1,1}$. Now, both actions 3 and 4 are ready, but we choose action 3 and then action 4. This results in the encoding given in the third row of Table 5. The canonical derivation of the derivation in Figure 11 is given in Figure 12. ∎

Figure 12: The canonical derivation of the derivation in Figure 11



An internal node is required by at most one action, since it is destroyed by that action. Thus, choosing the ready action that destroys the internal node with the smallest name is well defined. Furthermore, at each stage in the encoding process, at least one action is

Table 4: Association of names with pre-position numbers

v_1	v_2	v_3	v_4	v_5	v_6	$v_{1,0}$	$v_{1,1}$	$v_{2,0}$	$v_{2,1}$	$v_{3,0}$	$v_{3,1}$	$v_{4,0}$	$v_{4,1}$
4	1	0	0	0	0	0	3	0	2	0	0	0	0

Table 5: Encoding for the derivation in Figure 11

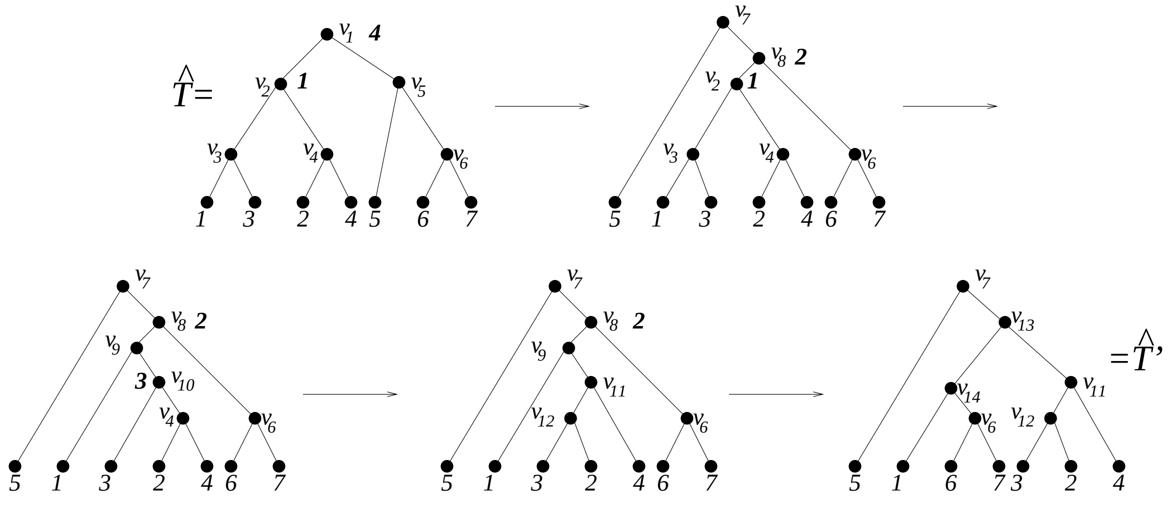
v_1	v_2	v_3	v_4	v_5	v_6	$v_{2,0}$	$v_{2,1}$	$v_{1,0}$	$v_{1,1}$	$v_{3,0}$	$v_{3,1}$	$v_{4,0}$	$v_{4,1}$
v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
4	1	0	0	0	0	0	2	0	3	0	0	0	0

ready. In particular, the first action of D among those that have not yet been performed is ready. This shows that one can reorder the actions of derivation D to form the canonical derivation D'.

Furthermore, the outcome of D' is identical to the outcome of D. If actions i and j of the original derivation D are ready at the same time while constructing D', then these actions do not require a common node, since each node is required by at most one action. Furthermore, neither action requires a node that exists as a result of the other, since they are ready at the same time. Robinson [15] proved that two rotations around two edges that do not share a common node can be performed in either order without affecting the outcome. This proves that the outcome of D' equals the outcome of D.

Next, we explain how the canonical derivation D' can be reconstructed (decoded) when $\hat{T}$ and the encoding are given. The decoding procedure mimics the behaviour of the encoding procedure. The encoding is simply a list of nonnegative integers, as in the third line of Table 5. We associate names $v_1, \ldots, v_{n+2m}$ with these integers in order. The first n names are those of the initial tree $\hat{T}$. Inspecting the parent-child pairs in $\hat{T}$ identifies which actions are ready. No two actions sharing a node can be ready simultaneously. We then apply the rule that destroys the node with the smallest name. This will obviously be the first action of D'. Application of this rule will create internal nodes v_{n+1} and v_{n+2}, corresponding with the $(n+1)$-th and $(n+2)$-th entry of the code. Continuing in this manner, we can reconstruct D'.

Figure 13: The decoding procedure



Example 5 Next to the nodes of Figure 13 we have written the corresponding entries from the encoding. In order not to overload the figure, the entries that equal zero are not shown. As can be seen, rules 1 and 4 can be applied to the initial tree. Because rule 4 destroys the node with the smallest name i.e. v_1, we apply this rule thus creating the nodes v_7 and v_8. Now we can only apply rule 1, yielding the third tree. After two more actions, we arrive at $\hat{T}'$. ∎

Lemma 6 *The number of trees within distance m from any binary coupling tree $T \in \mathcal{T}_n$ is at most $\binom{n+2m}{m}4^m$.*

Proof: Each application of a rule to an ordered tree corresponds to traversal of exactly one edge in G_n, and each edge traversal can be achieved by applying one of these rules. Hence we consider the number of trees that are reachable from T via derivations of length m.

Using this technique, every tree $\hat{T}'$ with $d(\hat{T}, \hat{T}') = m$ can be encoded by an array (code) of length $n + 2m$. Since a code of length $n + 2m$ has exactly m nonzero entries, and each nonzero entry lies in $\{1, 2, 3, 4\}$, the number of codes $|C(n, m)|$ of length $n + 2m$

is bounded by

$$|C(n,m)| \le \binom{n+2m}{m} 4^m.$$

The number of trees at distance m from any given tree is bounded by this number. We can even say more: the number of trees *within* distance m from any given tree is bounded by the same number. Indeed, if $d(T,T') = m-2l$, then there is a derivation of length m carrying T into T'; one only needs to go back and forth between T' and its predecessor on the path of length $m-2l$. If $d(T,T') = m-2l-1$, then one can construct a derivation of length $m-2l$ by adding a detour through the common neighbour of T' and its predecessor on a path of length $m-2l-1$, since every edge in G_n lies on a triangle (see Figure 2). The argument for $m-2l$ then applies. The bound holds also when $m=1$, since $1+2(n-1) < 4(n+2)$.

□

Theorem 7 *For $n > 1$, the diameter of G_n satisfies*

$$\operatorname{diam}(G_n) > \frac{1}{4}\lg(n!) > \frac{1}{4} n \lg(\frac{n}{e}).$$

Proof: Let $D = \operatorname{diam}(G_n)$. By Lemma 6,

$$\binom{n+2D}{D} 4^D \ge (2n-1)!! = \frac{n!}{2^n}\binom{2n}{n}. \tag{9}$$

Using asymptotic expansions (Stirling's formula),

$$\frac{2^{2n}}{\sqrt{\pi n}} > \binom{2n}{n} > \frac{2^{2n}}{\sqrt{\pi n}}\left(1-\frac{1}{8n}\right). \tag{10}$$

From (10) and $\binom{n+2D}{D} < \binom{n+2D}{n/2+D}$,

$$2^{4D} > n!\left(1-\frac{1}{8n}\right)\sqrt{\frac{1}{2}+\frac{D}{n}}.$$

For $n \ge 5$, the elementary lower bound (Moore bound) $D \ge n(\ln(2n)-1)/\ln(2n)$ in (8) now yields $2^{4D} > n!$. One can check directly that this bound also holds for $2 \le n \le 4$. □

Remark 8 One slightly improves the lower bound from Theorem 7 when bounding the left side of (9), for $n > 0$ and $D > 1$, by

$$\frac{2^{n+2D}}{2n} > \binom{n+2D}{D}, \tag{11}$$

and the right side of (9) by

$$\frac{(2n)!}{2^n n!} \geq \sqrt{2}\left(\frac{2n}{e}\right)^n \left(1-\frac{1}{24n}\right). \tag{12}$$

instead of by (10). Inequality (12) is proved using Stirling's formula. ∎

Combining Theorem 3 and Theorem 7 yields Theorem 1.

Acknowledgements

The authors would like to thank the referee for many useful suggestions and constructive criticism.

References

[1] B. Bollobás, Extremal graph theory. London Mathematical Society Monographs, 11. (Academic Press, 1978).

[2] F. Buckley and F. Harary, Distance in Graphs (Addison–Wesley, 1990).

[3] K. Culik II and D. Wood, A note on some tree similarity measures, Inform. Process. Lett. 15 (1982) 39–42.

[4] W. H. Day, Properties of the nearest neighbor interchange metric for trees of small size, J. Theor. Biol. 101 (1983) 275–288.

[5] V. Fack, S. Lievens and J. Van der Jeugt, On rotation distance between binary coupling trees and applications for $3nj$-coefficients, Comput. Phys. Commun. 119 (1999) 99–114.

[6] V. Fack, S. N. Pitre and J. Van der Jeugt, New efficient programs to calculate general recoupling coefficients. Part I: Generation of a summation formula, Comput. Phys. Commun. 83 (1994) 275–292.

[7] R. L. Graham, D. E. Knuth and O. Patashnik, Concrete Mathematics, A Foundation for Computer Science (Addison–Wesley, 1995).

[8] J. Jarvis, J. Luedeman and D. Shier, Comments on computing the similarity of binary trees, J. Theor. Biol. 100 (1983) 427–433.

[9] D. E. Knuth, Sorting and Searching, Volume 3 of The Art of Computer Programming (Addison–Wesley, 1973).

[10] D. E. Knuth, Fundamental Algorithms, Volume 1 of The Art of Computer Programming (Addison–Wesley, 1997).

[11] M. Li, J. Tromp and L. Zhang, On the nearest neighbour interchange distance between evolutionary trees, J. Theor. Biol. 182 (1996) 463–467.

[12] F. Luccio and L. Pagli, On the upper bound on the rotation distance of binary trees, Inform. Process. Lett. 31 (1989) 57–60.

[13] E. Mäkinen, On the rotation distance of binary trees, Inform. Process. Lett. 26 (1987) 271–272.

[14] B. McKay. nauty. `http://cs.anu.edu.au/people/bdm/nauty/`.

[15] D. Robinson, Comparison of labeled trees with valency three, J. Comb. Theory 11 (1971) 105–119.

[16] R. O. Rogers and R. D. Dutton, On distance in the rotation graph of binary trees, Congr. Numer. 120 (1996) 103–113.

[17] D. D. Sleator, R. E. Tarjan and W. P. Thurston, Short encodings of evolving structures, SIAM J. Disc. Math. 5 (1982) 428–450.

[18] D. D. Sleator, R. E. Tarjan and W. P. Thurston, Rotation distance, triangulations and hyperbolic geometry, J. Amer. Math. Soc. 1 (1988) 647–681.

[19] N. J. Sloane, On-line encyclopedia of integer sequences, `http://www.research.att.com/~njas/sequences/index.html`.

[20] M. S. Waterman and T. F. Smith, On the similarity of dendrograms, J. Theor. Biol. 73 (1978) 789–800.