ROTATION DISTANCE

Daniel D. Sleator
Robert E. Tarjan
William P. Thurston

July, 1985

CS-TR-003-85

Rotation Distance

Daniel D. Sleator
AT&T Bell Laboratories
Murray Hill, NJ 07974

Robert E. Tarjan
Computer Science Department
Princeton University
Princeton, NJ 08544

and

AT&T Bell Laboratories
Murray Hill, NJ 07974

William P. Thurston
Mathematics Department
Princeton University
Princeton, NJ 08544

July, 1985

ABSTRACT

In this note we summarize our recent results on *rotation distance*, a distance measure on binary trees with computer science applications. Our main result is that the maximum rotation distance between any two n-node binary trees is at most $2n-6$ for $n \geqslant 11$, and this bound is tight for infinitely many n.

Rotation Distance

A *rotation* is a local transformation on a binary tree that changes the depths of certain nodes but preserves the symmetric order of the nodes. (See Figure 1.) A rotation takes $O(1)$ time on any standard representation of a binary tree. Rotations are the operations used to rebalance binary search trees [3,6]; thus they play a fundamental role in data structures.

[Figure 1]

Rotations also impose a mathematical structure on the set of all n-node binary trees. Let R_n, the *rotation graph*, be the undirected graph whose vertices are the n-node binary trees, such that two trees are adjacent if and only if one can be obtained from the other by a single rotation. Let $d(T_1, T_2)$, the *rotation distance* between trees T_1 and T_2, be the distance between T_1 and T_2 in R_n, i.e. the minimum number of rotations needed to transform T_1 into T_2 or vice-versa. This note summarizes our recent work on rotation distance. Further details and proofs will appear in [5].

We formulate two fundamental questions about rotation distance:

Problem 1. Let d_n be the diameter of R_n, i.e. the minimum number of rotations that suffice to transform any n-node binary tree into any other. What is d_n?

Problem 2. Devise a polynomial-time algorithm that, given any two n-node binary trees T_1 and T_2, computes $d(T_1, T_2)$.

Our results provide an almost-complete solution to Problem 1 and an approximate solution to Problem 2. Concerning Problem 1, we prove:

Theorem 1. $d_n \leqslant 2n-6$ for all $n \geqslant 11$.

Theorem 2. $d_n = 2n-6$ for infinitely many n.

We conjecture but cannot yet prove that $d = 2n-6$ for all $n \geqslant 11$. However, we believe that an extension of our methods will establish this. We have computed the exact value of d_n for $n \leqslant 16$.

(See Figure 2.) These results show that $d_n = 2n-6$ for $11 \leq n \leq 16$.

[Figure 2]

Concerning Problem 2, we exhibit a linear-time algorithm that will estimate $d(T_1, T_2)$ to within a factor of two. Coming closer than a factor of two in general seems hard; however, our methods allow the exact computation of $d(T_1, T_2)$ in various special cases.

There has been very little previous work on rotation distance. To our knowledge the only published work is by Culik and Wood [1], who defined the concept and showed that $d_n \leq 2n-2$ for all n. Leighton (private communication) showed that $d_n \geq 7n/4 - O(1)$ for infinitely many n.

The original definition of rotation distance is not so easy to study. Thus it is advantageous to transform it into something more amenable. The binary trees are counted by the Catalan numbers [2] as are many other mathematical objects, including triangulations of a polygon. It is these with which we shall work. The n-vertex binary trees are in 1-1 correspondence with the triangulations of an $n+2$-gon if rotationally equivalent triangulations are regarded as distinct. Furthermore, rotation on binary trees corresponds to the *diagonal flip* operation on triangulations, in which we remove a diagonal, causing two triangles to merge into a quadrilateral, and replace it with the other diagonal of the quadrilateral. (See Figure 3.) Rotation distance on binary trees corresponds to flip distance on triangulations; the *flip distance* $f(T_1, T_2)$ between two triangulations T_1 and T_2 of an n-gon is the minimum number of flips necessary to transform T_1 into T_2 (or vice-versa). In the triangulation setting, Problems 1 and 2 become:

Problem 1': Determine $f_n = \max\{f(T_1, T_2) \mid T_1 \text{ and } T_2 \text{ are triangulations of an } n\text{-gon}\}$.

Problem 2': Devise a polynomial-time algorithm to compute $f(T_1, T_2)$ for any triangulations T_1 and T_2.

[Figure 3]

We summarize our results on triangulations.

Theorem 1'. $f_n \leqslant 2n-10$ for all $n \geqslant 13$.

Proof. Any triangulation of an n-gon has $n-3$ diagonals. Given any vertex x of initial degree $d(x) < n-3$, we can increase $d(x)$ by one by a suitable diagonal flip. Thus in $n-3-d(x)$ flips we can produce the triangulation all of whose diagonals have one end at x. It follows that given any two triangulations T_1 and T_2 we can convert T_1 into T_2 in $2n-6-d_1(x) - d_2(x)$ flips, where x is any vertex, of degree $d_1(x)$ in T_1 and degree $d_2(x)$ in T_2. A little algebra shows that if $n \geqslant 13$, there is a vertex x such that $d_1(x) + d_2(x) \geqslant 4$. The theorem follows. □

Theorem 2'. $f_n = 2n-10$ for infinitely many n.

The proof of Theorem 2' is our most interesting and complicated result. It uses a second transformation of the problem, to triangulating a polyhedron, and relies on volumetric arguments in hyperbolic space.

Lemma 1. If T_1 and T_2 are any two triangulations having a common diagonal e, then any minimum-length sequence of flips from T_1 to T_2 leaves e alone; indeed any flip sequence from T_1 to T_2 that flips e uses at least two more flips than the minimum number.

Lemma 2. If T_1 and T_2 are any two triangulations with no common diagonals but some diagonal e of T_1 can be converted into a diagonal e' of T_2 in one flip, then there is a shortest flip sequence from T_1 to T_2 that first flips e to e'.

A further result along the lines of Lemmas 1 and 2 concerning diagonals fixable in two flips can be proved. However, such results seem to be of no help in solving Problem 2', because there are pairs of triangulations T_1 and T_2 such that fixing even a single diagonal requires $\Omega(n)$ flips. On the other hand, Lemma 1 allows us to estimate $f(T_1, T_2)$ to within a constant factor:

Theorem 3. Let $g(T_1, T_2)$ be the number of diagonals in T_1 that are not in T_2. Then $g(T_1, T_2) \leqslant f(T_1, T_2) \leqslant 2g(T_1, T_2)$.

We close by mentioning another problem having to do with rotations that arises in the study of self-adjusting search trees [4,7]. A *turn* is a pair of rotations as illustrated in Figure 4.

[Figure 4]

Problem 3. Starting from an arbitrary n-node binary tree T, what is the maximum number of right turns that can be made before no more are possible?

We conjecture that the maximum number of right turns is $O(n)$, but can only prove $O(n \log n)$. Note that, starting from an arbitrary tree, the maximum number of right rotations that can be made is exactly $\binom{n}{2}$.

References

[1] K. Culik II and D. Wood, "A note on some tree similarity measures," Info. Process. Lett. 15 (1982), 39-42.

[2] D. E. Knuth, *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, Second Edition, Addison-Wesley, Reading, MA, 1973.

[3] D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, Addison-Wesley, Reading, MA, 1973.

[4] D. D. Sleator and R. E. Tarjan, "Self-Adjusting binary search trees," *J. Assoc. Comput. Mach.* 32 (1985), to appear.

[5] D. D. Sleator, R. E. Tarjan, and W. P. Thurston, "Rotation distance, triangulations, and hyperbolic geometry," to appear.

[6] R. E. Tarjan, *Data Structures and Network Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983.

[7] R. E. Tarjan, "Sequential access in splay trees takes linear time," *Combinatorica*, to appear.

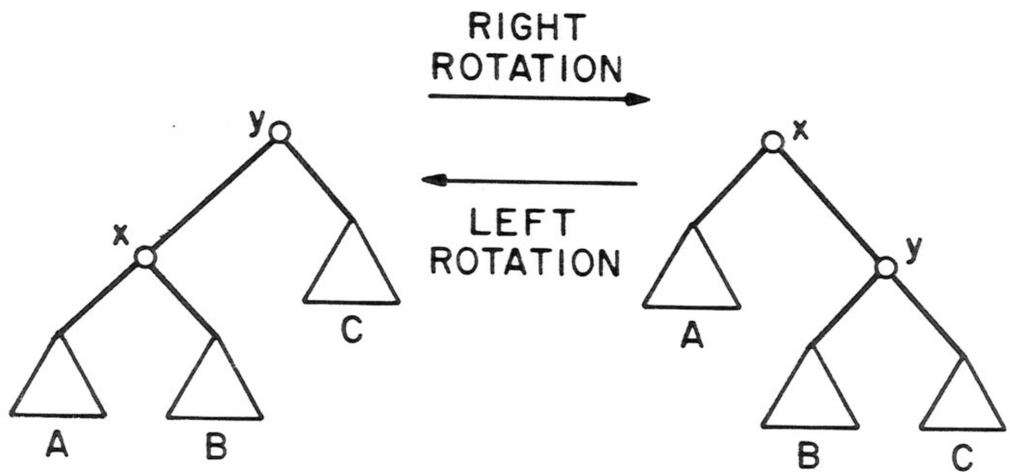

Figure 1. A rotation in a binary tree. Triangles denote subtrees. The tree shown could be part of a larger tree.

n	d_n
1	0
2	1
3	2
4	4
5	5
6	7
7	9
8	11
9	12
10	15
11	16
12	18
13	20
14	22
15	24
16	26

Figure 2. Values of d_n for small n.

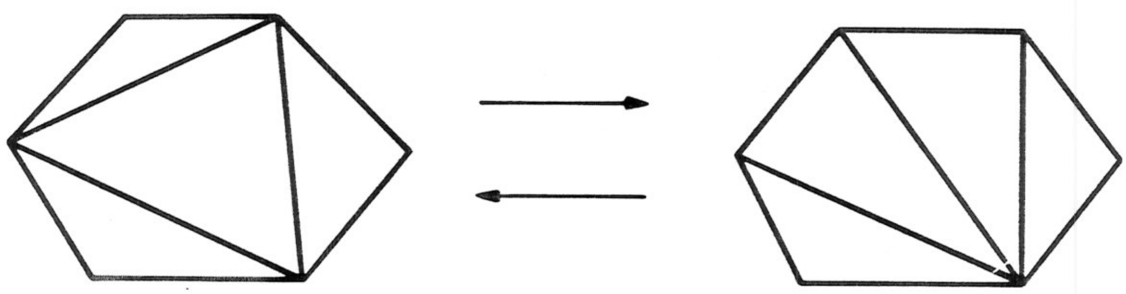

Figure 3. A diagonal flip in a triangulation.

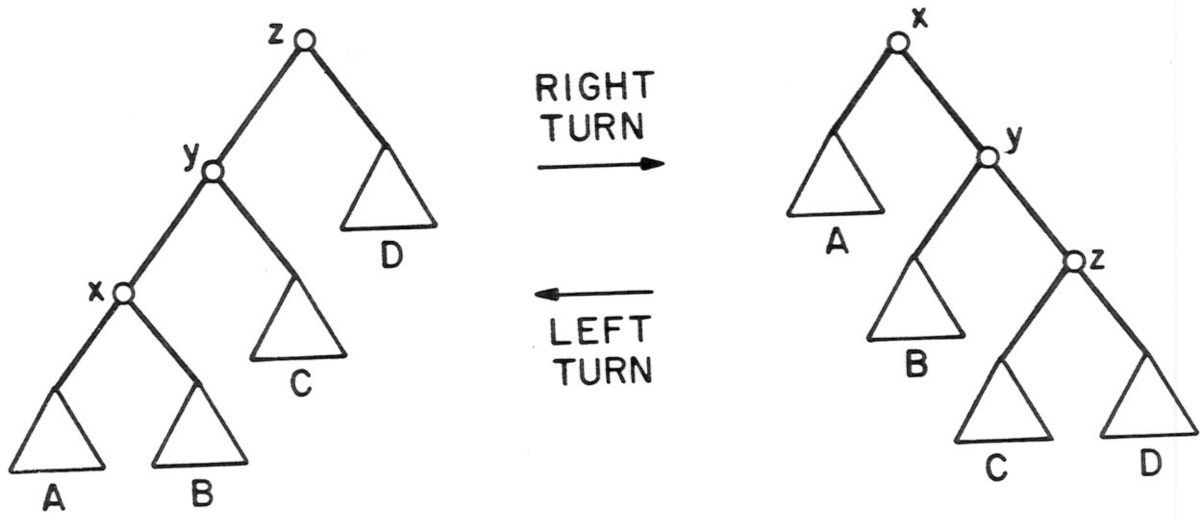

Figure 4. A turn on a binary tree.