

# TOWARD A LANGUAGE THEORETIC PROOF OF THE FOUR COLOR THEOREM

BOBBE COOPER, ERIC ROWLAND, AND DORON ZEILBERGER

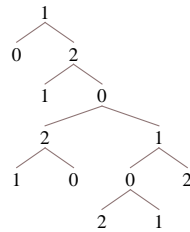
ABSTRACT. This paper considers the problem of showing that every pair of binary trees with the same number of leaves parses a common word under a certain simple grammar. We enumerate the common parse words for several infinite families of tree pairs and discuss several ways to reduce the problem of finding a parse word for a pair of trees to that for a smaller pair. The statement that every pair of trees has a common parse word is equivalent to the statement that every planar graph is four-colorable, so the results are a step toward a language theoretic proof of the four color theorem.

## 1. INTRODUCTION

Let  $G$  be the context-free grammar with start symbols  $0, 1, 2$  and formation rules  $0 \rightarrow 12, 0 \rightarrow 21, 1 \rightarrow 02, 1 \rightarrow 20, 2 \rightarrow 01, 2 \rightarrow 10$ . An  $n$ -leaf tree  $T$  *parses* a length- $n$  word  $w$  on  $\{0, 1, 2\}$  if  $T$  is a valid derivation tree for  $w$  under the grammar  $G$ ; that is, there is a labeling of the vertices of  $T$  compatible with the formation rules such that the leaves of  $T$ , from left to right, are labeled with the letters of  $w$ . For example, the tree



parses the word 0110212, as can be seen by this labeling:



Note that labeling the leaves of a tree uniquely determines a labeling of the internal vertices under  $G$  if a valid labeling exists.

---

*Date:* October 19, 2011.

*2010 Mathematics Subject Classification.* 05A05, 05C15, 68R15, 68Q42.

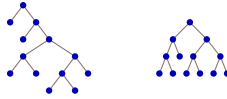
*Key words and phrases.* four color theorem, parse words.

The second author was supported in part by NSF grant DMS-0239996; the third author was supported in part by NSF grant DMS-0901226.

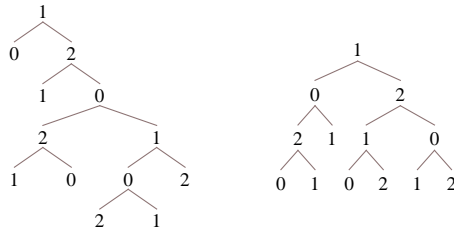
Certainly  $G$  is *ambiguous* — there exist distinct trees that parse the same word; for example, the trees



both parse 010. To take a somewhat larger example, the trees



both parse the word 0110212:



However, something much stronger can be said about this grammar.

**Theorem 1.** *The grammar  $G$  is totally ambiguous.*

That is, every pair of derivation trees with the same number of leaves has at least one word that they both parse. Kauffman [4] proved this theorem (in a slightly different form, as we describe below) by showing that it is equivalent to the four color theorem — the statement that every planar graph is four-colorable. The four color theorem was proved by Appel, Haken, and Koch [1, 2] using substantial computing resources. The hope of the present authors is that a direct proof of Theorem 1 will be shorter than the known proofs of the four color theorem, thereby providing a shorter proof of the four color theorem.

In this paper we describe first results in this direction. First we show that Theorem 1 is equivalent to Kauffman’s formulation. Section 3 determines explicit common parse words for several simple parameterized families of tree pairs. In Section 4 we establish existence of parse words for more general families. In Section 5 we enumerate the common parse words of a 3-parameter family of tree pairs. We conclude in Section 6 by discussing in more generality methods of reducing the problem of finding a common parse word for a pair of trees.

A *Mathematica* package [5] and a Maple package [6] that accompany this paper and facilitate the discovery of the results we present can be downloaded from the respective web sites of the second and third authors.

## 2. RELATIONSHIP TO THE CROSS PRODUCT

The set of possible derivation trees under  $G$  is the set of *binary trees* — trees in which each vertex has either 0 or 2 children. (All trees in the paper are rooted and ordered.)

Let  $|w|$  be the length of the word  $w$ , and let  $|w|_i$  be the number of occurrences of the letter  $i$  in  $w$ .

**Proposition 2.** *Let  $w$  be a word of length  $n$  on  $\{0, 1, 2\}$  and  $T$  an  $n$ -leaf binary tree that parses  $w$ . Then for some permutation  $(r, s, t)$  of  $(0, 1, 2)$ ,*

$$|w|_s \equiv |w|_t \not\equiv |w|_r \equiv |w| \pmod{2}.$$

*Moreover, the root of  $T$  receives the label  $r$  when parsing  $w$ .*

*Proof.* The congruence holds for the three words of length 1, and the derivation rules of  $G$  preserve it because all four terms change parity with each rule application.  $\square$

It follows that if the parities of  $|w|_0$ ,  $|w|_1$ , and  $|w|_2$  are equal then no tree parses  $w$  under  $G$ . If on the other hand the parity of  $|w|_r$  differs from the other two, then  $r$  is an invariant of  $w$  in the sense that any tree parsing  $w$  has its root labeled  $r$ .

Kauffman [4] formulated Theorem 1 not in terms of a grammar but in terms of the cross product on the standard unit vectors  $\hat{i}, \hat{j}, \hat{k}$  in  $\mathbb{R}^3$ . The cross product satisfies

$$\begin{aligned} \hat{i} \times \hat{j} &= (-\hat{i}) \times (-\hat{j}) = (-\hat{j}) \times \hat{i} = \hat{j} \times (-\hat{i}) = \hat{k}, \\ \hat{j} \times \hat{k} &= (-\hat{j}) \times (-\hat{k}) = (-\hat{k}) \times \hat{j} = \hat{k} \times (-\hat{j}) = \hat{i}, \\ \hat{k} \times \hat{i} &= (-\hat{k}) \times (-\hat{i}) = (-\hat{i}) \times \hat{k} = \hat{i} \times (-\hat{k}) = \hat{j}, \\ \hat{i} \times \hat{i} &= \hat{j} \times \hat{j} = \hat{k} \times \hat{k} = 0. \end{aligned}$$

Further, for every vector  $v \in \mathbb{R}^3$  we have  $v \times 0 = 0 \times v = 0$ . The cross product on  $\mathbb{R}^3$  is not associative, so in general the expression  $v_1 \times v_2 \times \cdots \times v_n$  is ambiguous; to evaluate it for a given tuple  $(v_1, v_2, \dots, v_n)$  we must choose an order in which to compute the  $n - 1$  cross products. Let us call such an order an  *$n$ -bracketing*. Kauffman showed that the four color theorem is equivalent to the statement that for every pair of  $n$ -bracketings there exists an  $n$ -tuple  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$  such that the two bracketings of  $v_1 \times v_2 \times \cdots \times v_n$  evaluate to the same nonzero vector.

We now develop tools to show that Theorem 1 is equivalent to Kauffman's statement. It is easy to see that  $n$ -bracketings are in bijection with  $n$ -leaf binary trees. Given an  $n$ -bracketing of  $v_1 \times v_2 \times \cdots \times v_n$ , one may label each internal leaf of the corresponding binary tree with the cross product of the labels of its children (in order). The condition that the bracketing does not evaluate to 0 is equivalent to the condition that the evaluation does not encounter the product  $\hat{i} \times \hat{i}$ ,  $\hat{j} \times \hat{j}$ , or  $\hat{k} \times \hat{k}$ , hence our formation rules for the grammar  $G$ .

Each  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$  possesses an invariant analogous to that of Proposition 2. To see what this invariant is, consider the quaternion group, whose elements are  $Q = \{1, i, j, k, -1, -i, -j, -k\}$  and whose binary operation  $\cdot$  satisfies

$$\begin{aligned} i \cdot j &= (-i) \cdot (-j) = (-j) \cdot i = j \cdot (-i) = k, \\ j \cdot k &= (-j) \cdot (-k) = (-k) \cdot j = k \cdot (-j) = i, \\ k \cdot i &= (-k) \cdot (-i) = (-i) \cdot k = i \cdot (-k) = j, \\ i \cdot i &= j \cdot j = k \cdot k = -1, \end{aligned}$$

as well as identities such as  $(-1) \cdot (-1) = 1$  and  $(-1) \cdot i = -i$  suggested by the notation. Consider  $\phi : \{\hat{i}, \hat{j}, \hat{k}, -\hat{i}, -\hat{j}, -\hat{k}\} \rightarrow \{i, j, k, -i, -j, -k\}$  mapping

$$\begin{aligned}\phi(\hat{i}) &= i, & \phi(-\hat{i}) &= -i, \\ \phi(\hat{j}) &= j, & \phi(-\hat{j}) &= -j, \\ \phi(\hat{k}) &= k, & \phi(-\hat{k}) &= -k.\end{aligned}$$

The map  $\phi$  is a “partial homomorphism” in the sense that  $\phi(v_1 \times v_2) = \phi(v_1) \cdot \phi(v_2)$  if  $v_1 \neq v_2$  and  $v_1 \neq -v_2$ . This property allows us to establish the following invariant.

**Proposition 3.** *Let  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$ , and choose a bracketing of  $v_1 \times v_2 \times \dots \times v_n$  that does not evaluate to the zero vector. Then this bracketing evaluates to  $\phi^{-1}(\phi(v_1) \cdot \phi(v_2) \cdots \phi(v_n))$ .*

*Proof.* Since the bracketing of  $v_1 \times v_2 \times \dots \times v_n$  does not evaluate to 0, each of the  $n - 1$  cross products is an operation on two linearly independent vectors. Therefore we may simulate the evaluation of the bracketing in  $Q$  rather than in  $\mathbb{R}^3$ , because replacing  $\times$  with  $\cdot$  is consistent with  $\phi$ . Since  $\cdot$  is associative, the bracketing evaluates to  $\phi(v_1) \cdot \phi(v_2) \cdots \phi(v_n)$  in  $Q$ . Moreover, since we do not encounter 0 in  $\mathbb{R}^3$ , we do not encounter 1 or  $-1$  in  $Q$ ; in particular,  $\phi(v_1) \cdot \phi(v_2) \cdots \phi(v_n) \in \{i, j, k, -i, -j, -k\}$ , and therefore  $\phi^{-1}(\phi(v_1) \cdot \phi(v_2) \cdots \phi(v_n))$  exists.  $\square$

A result of Proposition 3 is that we can drop the condition that the vectors are the same, so the four color theorem is equivalent to the statement that for every pair of  $n$ -bracketings there exists an  $n$ -tuple  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$  such that the two bracketings of  $v_1 \times v_2 \times \dots \times v_n$  evaluate to nonzero vectors.

Consider the homomorphism  $\sigma : Q \rightarrow Q/\{1, -1\} \cong V$ , where  $V = \{e, 0, 1, 2\}$  is the Klein four-group,  $e = \{1, -1\}$  is the identity element,  $0 = \{i, -i\}$ ,  $1 = \{j, -j\}$ , and  $2 = \{k, -k\}$ . Let  $\tau : \{\hat{i}, \hat{j}, \hat{k}\} \rightarrow \{0, 1, 2\}$  be defined by  $\tau(v) = \sigma(\phi(v))$ . In other words,  $\tau$  removes the hat and forgets the sign. Since  $\sigma$  is a homomorphism, evaluating  $T_1$  and  $T_2$  at  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$  results in nonzero vectors if and only if  $\tau(v_1)\tau(v_2) \cdots \tau(v_n)$  is a parse word for  $T_1$  and  $T_2$ . Therefore, for  $n$ -leaf binary trees  $T_1$  and  $T_2$ , the  $n$ -tuples  $(v_1, v_2, \dots, v_n) \in \{\hat{i}, \hat{j}, \hat{k}\}^n$  that evaluate to nonzero vectors when bracketed by  $T_1$  and  $T_2$  are in bijection with words  $w \in \{0, 1, 2\}^n$  that are parsed by both  $T_1$  and  $T_2$ . It follows that Theorem 1 is equivalent to the four color theorem.

### 3. PARAMETERIZED FAMILIES

In this section we introduce several families of binary trees and enumerate the parse words of several pairs of these trees. First we establish some additional terminology.

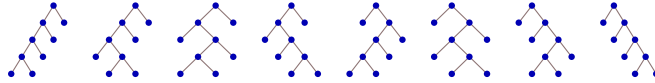
If  $T$  parses a word  $w$  on  $\{0, 1, 2\}$ , then  $T$  also parses all words obtained from  $w$  by permuting the letters in the alphabet. Let  $\text{ParseWords}(T_1, T_2)$  be the set of equivalence classes (under permutations) of words parsed by both trees  $T_1$  and  $T_2$ . We abuse notation slightly by writing a representative of each equivalence class. For example, it turns out that for the pair of 7-leaf trees mentioned in Section 1 there is only one equivalence class of parse words, so for those trees we write

$$\text{ParseWords}(T_1, T_2) = \{0110212\}.$$

Often we will take this representative to be the word in the equivalence class which is lexicographically first — words of the form 0 or  $0^k 1v$ . However, we will depart from this convention when convenient. The four color theorem is equivalent to the statement that for every pair of  $n$ -leaf binary trees  $T_1$  and  $T_2$  we have  $\text{ParseWords}(T_1, T_2) \neq \{\}$ .

The *level* of a vertex is its distance from the root. That is, the root lies on level 0, the root’s children lie on level 1, and so on.

A *path tree* is a binary tree with at most two vertices in each level. The 5-leaf path trees are as follows.

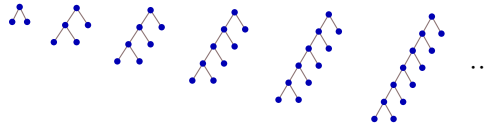


The two leaves on level  $n - 1$  in an  $n$ -leaf path tree are called the *bottom leaves*.

The set of  $n$ -leaf path trees is in trivial bijection to the set  $\{l, r\}^{n-2}$  of  $(n - 2)$ -length words on  $\{l, r\}$ : Since each level has at most two vertices, at most one vertex in each level has children, so we may form a word that records which child — left or right — has children at each level. We shall use this bijection to define several families of trees.

Because of their linear structure, path trees are simpler to work with than binary trees in general, so the emphasis of this paper is on path trees. Indeed, several infinite families of pairs of path trees can be shown to satisfy Theorem 1 directly and have only a few parse words. We take up this task now. Some of the proofs work by finding out where the local conditions imposed by the two trees force a unique labeling and then just working out the consequences, so in some cases it may be quicker to prove the theorem for yourself than to read the proof provided.

Let  $\text{LeftCombTree}(n)$  be the  $n$ -leaf path tree corresponding to the word  $l^{n-2}$ . The left comb trees for  $n \geq 2$  are pictured below.



Let  $\text{RightCombTree}(n)$  be the  $n$ -leaf path tree corresponding to  $r^{n-2}$ ;  $\text{RightCombTree}(n)$  is the left–right reflection of  $\text{LeftCombTree}(n)$ . We warm up with some *combinatorics*.

**Theorem 4.**  $\text{ParseWords}(\text{LeftCombTree}(n), \text{RightCombTree}(n)) =$

$$\begin{cases} \{01^{n-2}2\} & \text{if } n \geq 2 \text{ is even} \\ \{01^{n-2}0\} & \text{if } n \geq 3 \text{ is odd.} \end{cases}$$

*Proof.* We build a common parse word from left to right — up  $\text{LeftCombTree}(n)$  and down  $\text{RightCombTree}(n)$ . At every leaf, each tree will eliminate one possible label, so the parse word will turn out to exist and be unique.

The case  $n = 2$  can be established by testing all words of length 2, so let  $n \geq 3$ . Without loss of generality we may label the first two leaves 0 and 1. It follows from this that the root of  $\text{RightCombTree}(n)$  receives the label 1, the non-leaf (internal) vertex on the second level of  $\text{RightCombTree}(n)$  receives 2, and therefore the internal vertex on the third level of  $\text{RightCombTree}(n)$  receives 0. This implies

(from the right comb) that the third leaf cannot receive 0. However, from the left comb we find that the third leaf cannot receive 2. Therefore the third leaf receives 1. For the fourth leaf, the right comb precludes 2 and the left comb precludes 0, so the fourth leaf receives 1. Likewise all the way down the word through leaf  $n - 1$ . The internal vertex labels in each tree alternate between 0 and 2, except for the root which receives 1. If  $n$  is odd then the lowest internal vertex in the right comb receives 2, so that the last leaf receives 0; if  $n$  is even then this internal vertex receives 0, and the last leaf receives 2.  $\square$

Note from the proof of this theorem that the internal labels corresponding to a common parse word of  $\text{LeftCombTree}(n)$  and  $\text{RightCombTree}(n)$  will match (top to bottom) if  $n$  is odd, and will differ by the permutation which swaps 0 and 2 if  $n$  is even.

Let  $\text{LeftTurnTree}(m, n)$  be the  $(m + n)$ -leaf path tree corresponding to  $l^m r^{n-2}$ , and let  $\text{RightTurnTree}(m, n)$  be the tree corresponding to  $r^m l^{n-2}$ . Each of these trees is formed by “gluing” together two comb trees. For example,

$$\text{LeftTurnTree}(2, 3) = \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \bullet \quad \bullet \\ \diagup \quad \diagdown \quad \diagup \quad \diagdown \\ \bullet \quad \bullet \quad \bullet \quad \bullet \\ \diagup \quad \diagdown \quad \diagup \quad \diagdown \\ \bullet \quad \bullet \quad \bullet \quad \bullet \end{array} .$$

The following theorem is a special case of the general treatment of two turn trees given in Section 5.

**Theorem 5.** For  $m \geq 1$ ,

$$\begin{aligned} \text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(1, m + n - 1)) = \\ \begin{cases} \{001^{n-3}20^m, 021^{n-3}00^m\} & \text{if } n \geq 3 \text{ is odd} \\ \{021^{n-3}20^m, 001^{n-3}00^m\} & \text{if } n \geq 4 \text{ is even.} \end{cases} \end{aligned}$$

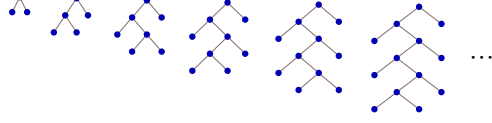
*Proof.* Without loss of generality, label the last leaf of each tree 0. The roots of the trees receive the same label, and thus the respective parents of the last leaf of each tree must receive 1 and 2 in some order, and the first leaf must be labeled 0. This implies that the last  $m$  leaves are labeled 0. There are (up to permutation of 1 and 2) three possible options for the labels of leaves  $n - 1$  and  $n$  (the bottom leaves of  $\text{LeftTurnTree}(m, n)$ ), namely 12, 10, and 01. Each of the first two options can be seen to yield a unique common parse word as given in the statement of the theorem. The third option, in which leaves  $n - 1$  and  $n$  are labeled 01, is not valid, since then the sibling of leaf  $n + 1$  in  $\text{RightTurnTree}(1, m + n - 1)$  is labeled 0, which contradicts leaf  $n + 1$  receiving 0.  $\square$

If  $w = w_1 w_2 \cdots w_m$  is a word of length  $m$  and  $x$  is a rational number whose denominator (in lowest terms) divides  $m$ , let

$$w^x = w^{\lfloor x \rfloor} w_1 w_2 \cdots w_{m \cdot (x - \lfloor x \rfloor)}$$

be the word consisting of repeated copies of  $w$  truncated at  $mx$  letters. For example,  $(lr)^{7/2} = lr l r l r l$ .

Let  $\text{LeftCrookedTree}(n)$  be the path tree corresponding to  $(lr)^{(n-2)/2}$ . The left crooked trees for  $n \geq 2$  are as follows.



Let  $\text{RightCrookedTree}(n)$  be the path tree corresponding to  $(rl)^{(n-2)/2}$  — the left-right reflection of  $\text{LeftCrookedTree}(n)$ .

The next two results determine the common parse words of a comb tree and the completely crooked trees of the same size. Let  $w^R$  be the left-right reversal of the word  $w$ . Let  $\text{mod}(n, 3)$  be the smallest nonnegative integer congruent to  $n$  modulo 3.

**Theorem 6.**  $\text{ParseWords}(\text{LeftCombTree}(n), \text{RightCrookedTree}(n)) =$

$$\begin{cases} \left\{ \text{mod}(1-n, 3) \left( (012)^{n/6} \right)^R (012)^{(n-2)/6} \right\} & \text{if } n \geq 2 \text{ is even} \\ \left\{ \text{mod}(1-n, 3) \left( (012)^{(n-3)/6} \right)^R (012)^{(n+1)/6} \right\} & \text{if } n \geq 3 \text{ is odd.} \end{cases}$$

*Proof.* One checks that for  $n = 2$  the set of equivalence classes of parse words is  $\{20\}$ .

Inductively, assume that  $\text{LeftCombTree}(n-1)$  and  $\text{RightCrookedTree}(n-1)$  parse the word claimed and that this is the only word they both parse (up to permutations of the alphabet). For even  $n-1$ , the two bottom leaves of  $\text{RightCrookedTree}(n-1)$  are leaves  $\frac{n-1}{2}$  and  $\frac{n+1}{2}$ . For odd  $n-1$ , they are leaves  $\frac{n}{2}$  and  $\frac{n+2}{2}$ . Observe that for even  $n-1$  the right bottom leaf of  $\text{RightCrookedTree}(n-1)$  receives 0, and for odd  $n-1$  the left bottom leaf of  $\text{RightCrookedTree}(n-1)$  receives 0. For  $n-1 \geq 4$  these are respectively the first and second of the two consecutive 0s in the parse word.

We attach  $\blacktriangle$  at the bottom of  $\text{RightCrookedTree}(n-1)$  to form  $\text{RightCrookedTree}(n)$  and insert  $\blacktriangle$  at the corresponding place in  $\text{LeftCombTree}(n-1)$  to form  $\text{LeftCombTree}(n)$ . Label the new bottom leaves of  $\text{RightCrookedTree}(n)$  12 if  $n-1$  is even and 21 if  $n-1$  is odd; we can label the corresponding leaves of  $\text{LeftCombTree}(n)$  the same by labeling their respective neighboring internal vertices 0 and 1 if  $n-1$  is even and 0 and 2 if  $n-1$  is odd. The permutation  $0 \rightarrow 2, 1 \rightarrow 0, 2 \rightarrow 1$  puts the new word in the form given in the theorem.

This process is reversible, so every parse word for  $n$  comes from a parse word for  $n-1$ .  $\square$

The next theorem follows immediately from the previous theorem by labeling  $\text{LeftCombTree}(n)$  and  $\text{RightCrookedTree}(n)$  with a common parse word and then attaching the root of each tree as the left leaf of  $\blacktriangle$ .

**Theorem 7.**  $\text{ParseWords}(\text{LeftCombTree}(n), \text{LeftCrookedTree}(n)) =$

$$\begin{cases} \left\{ \text{mod}(2-n, 3) \left( (012)^{(n-1)/6} \right)^R (012)^{(n-3)/6} \text{mod}(2-n, 3), \right. & \text{if } n \geq 3 \text{ is odd} \\ \left. \text{mod}(2-n, 3) \left( (012)^{(n-1)/6} \right)^R (012)^{(n-3)/6} \text{mod}(-n, 3) \right\} \\ \left\{ \text{mod}(2-n, 3) \left( (012)^{(n-4)/6} \right)^R (012)^{n/6} \text{mod}(2-n, 3), \right. & \text{if } n \geq 4 \text{ is even.} \\ \left. \text{mod}(2-n, 3) \left( (012)^{(n-4)/6} \right)^R (012)^{n/6} \text{mod}(-n, 3) \right\} \end{cases}$$

**Theorem 8.** For  $n \geq 2$ ,

$$|\text{ParseWords}(\text{LeftCrookedTree}(n), \text{RightCrookedTree}(n))| = 2^{\lfloor n/2 \rfloor - 1}.$$

*Proof.* The cases  $n = 2$  and  $n = 3$  are easily verified. In particular, every parse word of the 3-leaf pair consisting of



is of the form  $aba$  for  $a \neq b$ , where the two roots also get labeled  $b$ .

Let  $n$  be odd. Consider inductively extending  $\text{LeftCrookedTree}(n-2)$  and  $\text{RightCrookedTree}(n-2)$  by



respectively to obtain  $\text{LeftCrookedTree}(n)$  and  $\text{RightCrookedTree}(n)$ . Because the three new leaves are leaves  $(n-1)/2$ ,  $(n+1)/2$ , and  $(n+3)/2$  in both  $\text{LeftCrookedTree}(n)$  and  $\text{RightCrookedTree}(n)$ , every parse word

$$w_1 w_2 \cdots w_{(n-3)/2} b w_{(n+1)/2} \cdots w_{n-3} w_{n-2}$$

for the two  $(n-2)$ -leaf crooked trees can be extended to a parse word

$$w_1 w_2 \cdots w_{(n-3)/2} a b a w_{(n+1)/2} \cdots w_{n-3} w_{n-2}$$

for the two  $n$ -leaf crooked trees. Moreover, every parse word for the two  $n$ -leaf crooked trees can be obtained in this way. Since there are two choices for  $a$ , there are twice as many parse words for the  $n$ -leaf crooked trees as for the  $(n-2)$ -leaf crooked trees, which establishes the statement for odd  $n$ ; we see that  $w = w_1 w_2 \cdots w_{n-1} w_n$  is a parse word for  $\text{LeftCrookedTree}(n)$  and  $\text{RightCrookedTree}(n)$  if and only if  $w_i = w_{n+1-i} \neq w_{(n+1)/2}$  for  $1 \leq i \leq \frac{n-1}{2}$ .

Let  $n$  be even. Then every parse word of the  $n$ -leaf crooked trees can be obtained by extending a parse word  $w_1 w_2 \cdots w_{n/2-1} b w_{n/2+1} \cdots w_{n-2} w_{n-1}$  of the  $(n-1)$ -leaf crooked trees. Every parse word of  $\blacktriangle$  in which the root receives label  $b$  is of the form  $ac$ , where  $a \neq b$  and  $c \neq b$ , so there are twice as many parse words for the  $n$ -leaf crooked trees as for the  $(n-1)$ -leaf crooked trees, which establishes the statement for even  $n$ . Specifically,  $w = w_1 w_2 \cdots w_{n-1} w_n$  is a parse word for  $\text{LeftCrookedTree}(n)$  and  $\text{RightCrookedTree}(n)$  if and only if  $w_{n/2} \neq w_{n/2+1}$  and for some  $b \in \{0, 1, 2\}$  we have  $w_i = w_{n+1-i} \neq b$  for  $1 \leq i \leq \frac{n}{2} - 1$ .  $\square$

#### 4. GENERAL FAMILIES

Presumably explicit parse words can be found for various other parameterized families of tree pairs, but we now take a more general approach and establish results for tree pairs in which at least one of the trees does not come from a simple parameterized family. Some of these results will be used in Section 5. Note that, where stated, these results apply to not just path trees but binary trees in general.

**Proposition 9.** Let  $n \geq 3$ . If the  $i$ th leaf is a bottom leaf in two  $n$ -leaf path trees, then the trees both parse the word  $0^{k-1} 10^{n-k}$  for some  $2 \leq k \leq n-1$ .

For example, this proposition applies to the pair in Theorem 8 consisting of  $\text{LeftCrookedTree}(n)$  and  $\text{RightCrookedTree}(n)$ .

*Proof.* If  $i = 1$  then the second leaf is also a bottom leaf in both trees, so let  $k = 2$ ; similarly, if  $i = n$ , let  $k = n-1$ . If  $2 \leq i \leq n-1$ , let  $k = i$ . Labeling the  $k$ th leaf 1 and all other leaves 0 produces a valid labeling of both trees because the internal

vertices of the two trees on each level receive the same label, namely alternating between 2 and 1.  $\square$

We now give two propositions regarding extending a pair of binary trees by  $\blacktriangle$ .

**Proposition 10.** *Suppose  $T'_1$  and  $T'_2$  are  $n$ -leaf binary trees with a common parse word. Extend each tree by attaching  $\blacktriangle$  to leaf  $i$ , obtaining  $T_1$  and  $T_2$  respectively. Then*

$$|\text{ParseWords}(T_1, T_2)| = 2|\text{ParseWords}(T'_1, T'_2)|.$$

*In particular,  $T_1$  and  $T_2$  have a common parse word.*

*Proof.* Let  $w$  be a parse word of  $T'_1$  and  $T'_2$ . Without loss of generality we may assume that  $w_i = 0$ . Replacing  $w_i$  by 12 or 21 produces a word that both  $T_1$  and  $T_2$  parse, and every parse word for the pair arises uniquely in this way.  $\square$

In the next proposition we consider extending a tree  $T$  by inserting  $\blacktriangle$  into the tree at an internal vertex to “duplicate” a leaf. Fix  $i$ , and let  $S$  be the tree hanging from the sibling vertex of leaf  $i$ . Remove  $S$  from its position, attach  $\blacktriangle$  to the sibling of leaf  $i$ , and then reattach  $S$  to a leaf of the new  $\blacktriangle$  as follows. If leaf  $i$  is a left leaf, attach  $S$  to the right leaf of the new  $\blacktriangle$ ; if leaf  $i$  is a right leaf, attach  $S$  to the left leaf. Therefore if leaf  $i$  in  $T$  is a left leaf, then leaves  $i$  and  $i + 1$  in the extended tree are both left leaves, and if leaf  $i$  in  $T$  is a right leaf, then leaves  $i$  and  $i + 1$  in the extended tree are right leaves. We refer to this operation as *duplicating* leaf  $i$ .

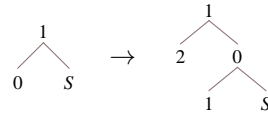
**Proposition 11.** *Suppose  $T'_1$  and  $T'_2$  are  $n$ -leaf binary trees with a common parse word. Extend  $T'_1$  by attaching  $\blacktriangle$  to leaf  $i$ , obtaining  $T_1$ . Extend  $T'_2$  to obtain  $T_2$  by duplicating leaf  $i$ . Then*

$$|\text{ParseWords}(T_1, T_2)| = |\text{ParseWords}(T'_1, T'_2)|.$$

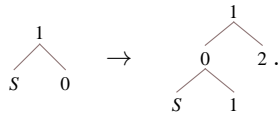
*In particular,  $T_1$  and  $T_2$  have a common parse word.*

*Proof.* Let  $w$  be a parse word of  $T'_1$  and  $T'_2$ . Without loss of generality we may assume that  $w_i = 0$  and that the parent of leaf  $i$  in  $T'_2$  receives the label 1.

If leaf  $i$  is a left leaf in  $T'_2$ , then  $T_2$  parses the word obtained by replacing  $w_i$  by 21 since duplicating leaf  $i$  in  $T'_2$  has the effect of the replacement



at the parent of leaf  $i$ , which preserves the labels of all other vertices. If leaf  $i$  is a right leaf in  $T'_2$ , then  $T_2$  parses the word obtained by replacing  $w_i$  by 12 since now the replacement is



Clearly  $T_1$  parses both of these words, so we have found a parse word for the pair. Moreover, every parse word of  $T_1$  and  $T_2$  arises uniquely in this way.  $\square$

In Section 3 we referred to the two leaves of maximal depth in a path tree as *bottom leaves*. In a general binary tree, a *bottom leaf* is a leaf whose sibling is also a leaf. It is clear that for  $n \geq 2$  every  $n$ -leaf binary tree has at least one pair of

bottom leaves, and every binary tree that is not a path tree has at least two pairs of bottom leaves. We use these facts in the next two theorems.

**Theorem 12.** *Let  $n \geq 2$ , and let  $T$  be an  $n$ -leaf binary tree. Let  $l$  be the level of leaf 1 in  $T$ . Then  $|\text{ParseWords}(T, \text{LeftCombTree}(n))| = 2^{l-1}$ .*

By symmetry, the analogous result holds for the right comb.

*Proof.* We work by induction on  $n$ . The only 2-leaf binary tree is  $\text{LeftCombTree}(2) = \blacktriangleright$ , which has only one parse word up to permutation of the alphabet.

Let  $T$  be an  $n$ -leaf binary tree. Then  $T$  has a pair of bottom leaves; suppose these are leaves  $i$  and  $i + 1$ . Remove these two leaves to obtain  $T'$ , which has  $n - 1$  leaves. If  $i = 1$ , then Proposition 10 gives twice as many parse words for  $T$  and  $\text{LeftCombTree}(n)$  as parse words for  $T'$  and  $\text{LeftCombTree}(n - 1)$ . If  $i > 1$ , then leaf  $i$  is a right leaf in  $\text{LeftCombTree}(n - 1)$ , so Proposition 11 gives the same number of parse words as for  $T'$  and  $\text{LeftCombTree}(n - 1)$ .  $\square$

Csar, Sengupta, and Suksompong [3] have recently provided a generalization of Theorem 12. They consider a partial ordering on the set of  $n$ -leaf binary trees arising from the rotation operation. They show that if  $T_1$  and  $T_2$  are  $n$ -leaf binary trees whose join exists under this partial ordering, then  $|\text{ParseWords}(T_1, T_2)|$  is a certain power of 2.

The following theorem is an extension of Theorem 12 to turn trees, although we lose the enumeration.

**Theorem 13.** *Let  $n \geq 4$ . Let  $T_1$  be an  $n$ -leaf binary tree and  $T_2$  an  $n$ -leaf left turn tree. Then  $T_1$  and  $T_2$  have a common parse word.*

*Proof.* We work by induction on  $n$ . For  $n = 4$  the result can be verified explicitly.

Now suppose that every  $(n - 1)$ -leaf binary tree has a common parse word with every  $(n - 1)$ -leaf left turn tree. Let  $T_1$  be an  $n$ -leaf binary tree, and let  $T_2$  be an  $n$ -leaf left turn tree. Then  $T_1$  has a pair of bottom leaves; suppose these are leaves  $i$  and  $i + 1$ .

First we consider the case where the  $i$ th leaf of  $T_2$  is the right bottom leaf. If  $T_1$  is a path tree, then  $T_1$  and  $T_2$  have a common parse word by Proposition 9. If  $T_1$  is not a path tree, then there is another pair of bottom leaves in  $T_1$ , so we may re-choose  $i$  if necessary so that the  $i$ th leaf of  $T_2$  is not the right bottom leaf.

Therefore we may assume that the  $i$ th leaf of  $T_2$  is not the right bottom leaf. Remove leaves  $i$  and  $i + 1$  from  $T_1$  to obtain  $T'_1$ , which has  $n - 1$  leaves and so has a common parse word with every  $(n - 1)$ -leaf left turn tree.

If the  $i$ th leaf of  $T_2$  is the left bottom leaf, then we can apply Proposition 10 to obtain a common parse word for  $T_1$  and  $T_2$ . Otherwise, leaves  $i$  and  $i + 1$  occur on consecutive levels in  $T_2$ , so Proposition 11 applies.  $\square$

## 5. A PAIR OF TURN TREES

In this section we give three theorems that collectively determine the number of parse words of  $\text{LeftTurnTree}(m, n)$  and  $\text{RightTurnTree}(k, m + n - k)$ . Note that by Theorem 13 the number of parse words is nonzero.

**Theorem 14.** *For  $m \geq 1$ ,  $k \geq 1$ , and  $\max(2, k - m + 2) \leq n \leq k$ ,*

$$|\text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(k, m + n - k))| = 1.$$

*Proof.* The bottom leaves of  $\text{LeftTurnTree}(m, n)$  (which are leaves  $n - 1$  and  $n$ ) correspond to leaves which are on consecutive levels in  $\text{RightTurnTree}(k, m + n - k)$ , so we can apply Proposition 11 to see that

$$\begin{aligned} & |\text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(k, m + n - k))| \\ &= |\text{ParseWords}(\text{LeftTurnTree}(m, n - 1), \text{RightTurnTree}(k - 1, m + n - k))|. \end{aligned}$$

Now, our hypothesis applies to this new, smaller tree pair, so we may continue reducing in the same way until we have reduced the right comb in the left turn tree entirely away. At this point, we are considering the trees  $\text{LeftTurnTree}(m, 2) = \text{LeftCombTree}(m + 2)$  and  $\text{RightTurnTree}(k - (n - 2), m + n - k)$ , which have a unique parse word class by Theorem 12.  $\square$

Let

$$a(m, k) = |\text{ParseWords}(\text{LeftTurnTree}(m, k + 1), \text{RightTurnTree}(k, m + 1))|.$$

By considering the left–right reflections of these two trees, we see that  $a(m, k) = a(k, m)$ . Theorem 15 determines the number of parse words of  $\text{LeftTurnTree}(m, n)$  and  $\text{RightTurnTree}(k, m + n - k)$  for  $n \geq k + 2$  in terms of  $a(m, k)$ , and Theorem 17 evaluates  $a(m, k)$ .

**Theorem 15.** *For  $m \geq 1$ ,  $k \geq 1$ , and  $n \geq k + 2$ ,*

$$|\text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(k, m + n - k))| = 2a(m, k).$$

*Proof.* If  $n > k + 2$ , then the bottom leaves of  $\text{LeftTurnTree}(m, n)$  correspond to leaves which are on consecutive levels in  $\text{RightTurnTree}(k, m + n - k)$ , so we can apply Proposition 11 to see that

$$\begin{aligned} & |\text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(k, m + n - k))| \\ &= |\text{ParseWords}(\text{LeftTurnTree}(m, n - 1), \text{RightTurnTree}(k, m + n - k - 1))|. \end{aligned}$$

If our hypothesis applies to this new, smaller tree pair, we may continue reducing in exactly the same way until we reach  $\text{LeftTurnTree}(m, k + 2)$  and  $\text{RightTurnTree}(k, m + 2)$ . Leaves  $k$  and  $k + 1$  are bottom leaves in both these trees, so by Proposition 10 we have

$$\begin{aligned} & |\text{ParseWords}(\text{LeftTurnTree}(m, n), \text{RightTurnTree}(k, m + n - k))| \\ &= 2|\text{ParseWords}(\text{LeftTurnTree}(m, k + 1), \text{RightTurnTree}(k, m + 1))|. \quad \square \end{aligned}$$

For the final result concerning the number of parse words of two turn trees, it turns out to be convenient to focus on the labels of the internal vertices rather than of the leaves. We form a word consisting of the internal vertex labels of a labeled path tree by reading these labels from top to bottom.

A word on  $\{0, 1, 2\}$  is *alternating* if no two consecutive letters are equal. If the internal vertices of a path tree are labeled with  $w$ , then the labeling can be extended to a parse word for the tree precisely when  $w$  is alternating. Therefore it will be important to know the sizes of certain sets of alternating words. Let  $A_m$  be the set of length- $m$  alternating words of the form  $0v_2 \cdots v_m$ , where  $v_2, v_m \in \{1, 2\}$ . Let  $B_m$  be the set of length- $m$  alternating words of the form  $0v_2 \cdots v_m$ , where  $v_2 \in \{1, 2\}$  and  $v_m \in \{0, 2\}$ .

**Proposition 16.** *For  $m \geq 2$ ,  $|A_m| = (2^m + 2(-1)^m)/3$  and  $|B_m| = (2^m - (-1)^m)/3$ .*

*Proof.* Let  $a_i(m)$  be the number of length- $m$  alternating words on  $\{0, 1, 2\}$  beginning with 01 and ending with  $\text{mod}(i, 3)$ . Then

$$\begin{aligned}
a_i(m) &= a_{i+1}(m-1) + a_{i+2}(m-1) \\
&= a_{i+2}(m-2) + 2a_{i+3}(m-2) + a_{i+4}(m-2) \\
&= a_{i+3}(m-3) + 3a_{i+4}(m-3) + 3a_{i+5}(m-3) + a_{i+6}(m-3) \\
&\vdots \\
&= \sum_{j=0}^n \binom{n}{j} a_{i+n+j}(m-n) \\
&\vdots \\
&= \sum_{j=0}^{m-2} \binom{m-2}{j} a_{i+m-2+j}(2) \\
&= \sum_{j \equiv -(i+m) \pmod{3}} \binom{m-2}{j}
\end{aligned}$$

since  $a_0(2) = a_2(2) = 0$  and  $a_1(2) = 1$ . Therefore

$$a_0(m) = \sum_{j \equiv -m \pmod{3}} \binom{m-2}{j} = \frac{1}{3} (2^{m-2} + (-1)^{m-1}).$$

Since  $a_1(m)$  and  $a_2(m)$  also count alternating words of the forms  $02 \cdots 2$  and  $02 \cdots 1$  respectively, we have

$$|A_m| = 2a_1(m) + 2a_2(m) = 2(2^{m-2} - a_0(m)) = \frac{1}{3} (2^m + 2(-1)^m).$$

Similarly,

$$|B_m| = 2a_0(m) + a_1(m) + a_2(m) = a_0(m) + 2^{m-2} = \frac{1}{3} (2^m - (-1)^m). \quad \square$$

Next we provide a simple recurrence satisfied by  $a(m, k)$ . Unfortunately, we do not know a correspondingly simple proof.

**Theorem 17.** For  $m \geq 1$  and  $k \geq 1$ ,

$$a(m+3, k) - 2a(m+2, k) - a(m+1, k) + 2a(m, k) = 0.$$

Initial conditions that suffice to completely determine  $a(m, k)$  from this recurrence are  $a(1, 1) = 1$ ,  $a(1, 2) = 1$ ,  $a(1, 3) = 1$ ,  $a(2, 2) = 4$ ,  $a(2, 3) = 5$ , and  $a(3, 3) = 3$ . The particular solution can be written as the matrix product

$$a(m, k) = \frac{1}{4} \begin{pmatrix} 2/3 \cdot 2^m \\ 1 \\ 5/3 \cdot (-1)^m \end{pmatrix}^\top \begin{pmatrix} 1/2 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1/5 \end{pmatrix} \begin{pmatrix} 2/3 \cdot 2^k \\ 1 \\ 5/3 \cdot (-1)^k \end{pmatrix}.$$

*Proof.* Let  $m \geq 2$  and  $k \geq 2$ , and let  $A_m$  and  $B_m$  be as above. Let

$$\begin{aligned} 1_m(w) &= \left\{ ((01)^{m/2}w, w(10)^{m/2}) \right\}, \\ 2_m(w) &= \left\{ ((02)^{m/2}w, w(20)^{m/2}) \right\}, \\ A_m(w) &= \{(vw, wv) : v \in A_m\}, \\ B_m((01)^{k/2}) &= \{(v(10)^{k/2}, (01)^{k/2}v) : v \in B_m\}. \end{aligned}$$

We consider the set of pairs  $(L, R)$  of length- $(m+k)$  (alternating) words such that  $R = 01 \cdots$  and such that respectively labeling the internal vertices of  $\text{LeftTurnTree}(m, k+1)$  and  $\text{RightTurnTree}(k, m+1)$  with  $L$  and  $R$  produces a parse word for the pair. Such pairs  $(L, R)$  are in bijection with equivalence classes of parse words for this tree pair as follows. The internal vertex labels of a path tree determine the labels of all leaves except the bottom leaves. Since  $\text{LeftTurnTree}(m, k+1)$  and  $\text{RightTurnTree}(k, m+1)$  do not share both bottom leaves, labeling the internal vertices with the pair  $(L, R)$  determines a unique parse word. We may choose representative parse words so that the internal vertex labels of  $\text{RightTurnTree}(k, m+1)$  begin with  $01$  (since the first two labels cannot be the same).

Let  $w = 01 \cdots$  be the length- $k$  prefix of  $R$ . Thus the internal vertices of the right comb of  $\text{RightTurnTree}(k, m+1)$  are labeled with letters from  $w$ , and the first letter of the parse word is 2. We show that if  $w$  contains all three letters then the set of pairs  $(L, R)$  is

$$\left\{ \begin{array}{ll} \{\} & \text{if } w \text{ ends in } 0 \text{ and } m \text{ is odd} \\ 1_m(w) \cup 2_m(w) & \text{if } w \text{ ends in } 0 \text{ and } m \text{ is even} \\ A_m(w) & \text{if } w \text{ ends in } 1 \text{ or } 2 \text{ and } m \text{ is odd} \\ A_m(w) \cup 2_m(w) & \text{if } w \text{ ends in } 1 \text{ and } m \text{ is even} \\ A_m(w) \cup 1_m(w) & \text{if } w \text{ ends in } 2 \text{ and } m \text{ is even,} \end{array} \right.$$

and if  $w = (01)^{k/2} = 0101 \cdots$  contains only two letters then this set is

$$\left\{ \begin{array}{ll} \{((01)^{(m+k)/2}, (01)^{(m+k)/2})\} & \text{if } w \text{ ends in } 0 \text{ and } m \text{ is odd} \\ 1_m(w) \cup 2_m(w) & \text{if } w \text{ ends in } 0 \text{ and } m \text{ is even} \\ A_m(w) \cup B_m(w) & \text{if } w \text{ ends in } 1 \text{ and } m \text{ is odd} \\ A_m(w) \cup B_m(w) \cup 2_m(w) & \text{if } w \text{ ends in } 1 \text{ and } m \text{ is even.} \end{array} \right.$$

To see this, first suppose that  $L = vw$  and  $R = wv$  for some  $v$ . Then  $v$  begins with 0, so  $w$  does not end in 0, and every  $v \in A_m$  produces a parse word.

Next suppose that  $L = v'w$  and  $R = wv$  for some  $v' \neq v$ . Then in fact  $v$  and  $v'$  differ in every position; in particular,  $v$  begins with some letter  $j \neq 0$ . Let  $i \in \{1, 2\}$  such that  $i \neq j$ . Then the final leaf receives the label  $i$  since it is a child of a 0 leaf in  $\text{LeftTurnTree}(m, k+1)$  and a child of a  $j$  leaf in  $\text{RightTurnTree}(k, m+1)$ . It follows that  $v = (j0)^{m/2}$  and  $v' = (0j)^{m/2}$ ; therefore  $m$  is even, and choosing  $j$  to be either 1 or 2 produces a parse word as long as it differs from the last letter of  $w$ .

Finally, suppose that  $L = vw'$  for some length- $k$  word  $w' \neq w$ . Then  $w$  and  $w'$  differ in every position; in particular,  $w'$  begins with 1, and it follows that  $w = (01)^{k/2}$  and  $w' = (10)^{k/2}$ . If  $w$  ends in 0, then  $L = R = (01)^{(m+k)/2}$ , yielding

the parse word  $2^k 0 2^m$ . If  $w$  ends in 1, then every  $v \in B_m$  produces a parse word, and  $R = wv$ .

Since we know the sizes of all these sets by Proposition 16, we can enumerate the set of internal word pairs  $(L, R)$  and obtain an expression for  $a(m, k)$ , which as expected is symmetric in  $m$  and  $k$ . For fixed  $k$  the expression is a linear combination of  $2^m$ , 1, and  $(-1)^m$ , so it satisfies the recurrence stated in the theorem, which can be written

$$(M - 2)(M - 1)(M + 1) a(m, k) = 0,$$

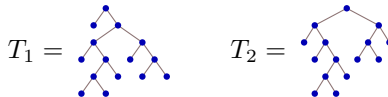
where  $M$  is the forward shift operator in the variable  $m$ .

When  $m = 1$  or  $k = 1$  one of the two trees is a comb tree, and by Theorem 12 we have  $a(m, k) = 1$ , which one checks is also what the general expression for  $a(m, k)$  gives upon setting  $m = 1$  or  $k = 1$ .  $\square$

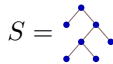
## 6. REDUCING A PAIR OF TREES

How might one proceed from the theorems of the previous sections to a proof that every two  $n$ -leaf path trees parse a common word? Here we introduce two notions of reducibility — ways to reduce the problem of finding a parse word for a pair of trees to finding parse words for smaller pairs — and give some related conjectures.

**6.1. Decomposable pairs.** Recall that if  $T_1$  and  $T_2$  are  $n$ -leaf trees such that leaves  $i$  and  $i + 1$  are siblings in both trees, then Proposition 10 reduces the problem of finding a parse word for  $T_1$  and  $T_2$  to the problem of finding a parse word for the pair of  $(n - 1)$ -leaf trees in which the common  $\blacktriangle$  has been removed. Our first observation is that there is nothing special about  $\blacktriangle$ ; if the two trees have any common branch system in the same position, then we can decompose the trees. For example, the 8-leaf trees



share the branch system



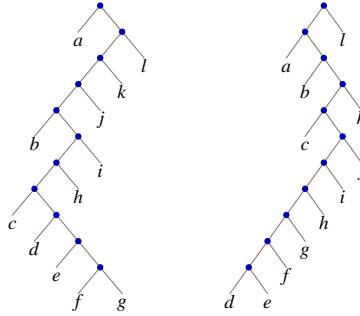
in the second through fifth leaves, which we may remove to obtain the 5-leaf trees



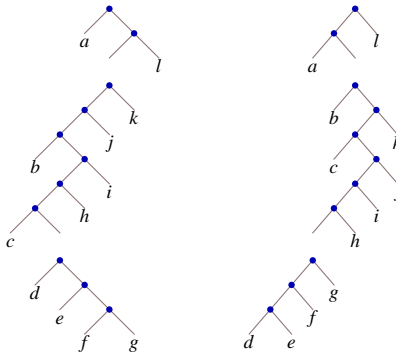
Given a common parse word  $w_1 w_2 w_3 w_4 w_5$  of this pair of 5-leaf trees, we can find a common parse word of the original pair of 8-leaf trees by taking any valid labeling of  $S$  and permuting the alphabet so that the root receives the label  $w_2$ .

In fact to decompose a pair of trees we only require a vertex in  $T_1$  with dangling subtree  $S_1$  and a vertex in  $T_2$  with dangling subtree  $S_2$  such that the leaves in  $S_1$

and  $S_2$  are the same. For example, there are two such vertex pairs in the tree pair



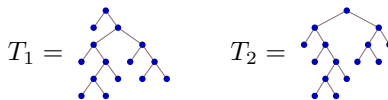
(where corresponding leaves have been given the same label). Breaking the trees at levels 2 and 8 as



produces the same partition  $\{\{a, l\}, \{b, c, h, i, j, k\}, \{d, e, f, g\}\}$  of the leaves in both trees. Thus, to find a parse word for a original pair it suffices to find parse words for the subtree pairs. Proposition 2 guarantees that we can reattach the subtrees consistently, since every binary tree that parses  $w$  receives the same label for its root when the leaves are labeled with the letters of  $w$ . Let us call a pair of path trees *indecomposable* if there is no such (nontrivial) decomposition.

The tree pair in Theorem 4 consisting of  $\text{LeftCombTree}(n)$  and  $\text{RightCombTree}(n)$  is indecomposable, as is the pair in Theorem 6 consisting of  $\text{LeftCombTree}(n)$  and  $\text{RightCrookedTree}(n)$ . On the other hand, breaking the trees  $\text{LeftCombTree}(n)$  and  $\text{LeftCrookedTree}(n)$  at level 1 shows that this pair is decomposable, and in this case the decomposition accounts for the non-uniqueness of the equivalence classes of words in Theorem 7.

The technique of decomposing trees is not limited to path trees. For example, the pair





mutually crooked. Experimental evidence suggests that in fact it suffices to consider pairs of mutually crooked trees.

**Conjecture 19.** *Let  $T'_1$  and  $T'_2$  be  $(n - 1)$ -leaf binary trees. Let  $1 \leq i \leq n - 1$ , and let  $T_1$  and  $T_2$  be the  $n$ -leaf trees obtained from  $T'_1$  and  $T'_2$  by duplicating leaf  $i$ . There exists a parse word  $w = w_1 \cdots w_n$  of  $T_1$  and  $T_2$  such that  $w_i = w_{i+1}$ .*

For example, in the pair discussed at the beginning of this subsection, leaves 5 and 6 are on consecutive levels in both trees, and these leaves receive the label 2. Note however that the parse word of  $T_1$  and  $T_2$  is not necessarily a simple extension of a parse word of  $T'_1$  and  $T'_2$ .

The trees `LeftCrookedTree( $n$ )` and `RightCrookedTree( $n$ )` (which we addressed in Theorem 8) are mutually crooked, but for  $n \geq 5$  no path tree is mutually crooked to `LeftCombTree( $n$ )`, since even a completely crooked tree has a pair of consecutive leaves that lie in consecutive levels. Theorems 4 and 5 provide additional examples of pairs that fail to be mutually crooked.

**6.3. Other conjectures.** To prove that every pair of  $n$ -leaf binary trees  $T_1$  and  $T_2$  has a parse word, it therefore suffices to consider indecomposable, weakly mutually crooked pairs of trees. In particular, we may assume that the leaves on level 1 in  $T_1$  and  $T_2$  are different, since if they are the same then the pair is decomposable at level 1 into smaller pairs.

**Theorem 20.** *Let  $n \geq 3$ , and let  $T_1$  and  $T_2$  be  $n$ -leaf path trees such that leaf 1 is on level 1 in  $T_1$  and leaf  $n$  is on level 1 in  $T_2$ . Then  $T_1$  and  $T_2$  have no parse word of the form  $01v1$ .*

*Proof.* For  $n = 3$  one checks that  $011$  is not a parse word for one of the two 3-leaf binary trees. Assume  $n \geq 4$ . Toward a contradiction, suppose that  $01v1$  is a common parse word for some  $v$ . Then the root of each tree receives the label 2, and the internal vertex on level 1 of  $T_1$  receives 1. Consider the children of this internal vertex. If the right child is a leaf, then it is leaf  $n$  and so receives 1, which is not a valid label because its parent is already labeled 1. If the left child is a leaf, then it is leaf 2 and so receives 1, which is also not a valid label.  $\square$

A similar argument shows that if there is a parse word of the form  $01v2$ , then leaf 2 of  $T_1$  is on level 2, and leaf  $n - 1$  of  $T_2$  is on level 2.

The following conjecture gives several statements that seem to be true and may be helpful in proving Theorem 1 for path trees directly.

**Conjecture 21.** *Let  $n \geq 4$ , and let  $T_1$  and  $T_2$  be  $n$ -leaf path trees such that leaf 1 is on level 1 in  $T_1$  and leaf  $n$  is on level 1 in  $T_2$ . Then we have the following.*

- *If  $T_1$  and  $T_2$  have no parse word of the form  $00v$  or  $v00$ , then they have a unique parse word (up to permutation of alphabet).*
- *If  $T_1$  and  $T_2$  have no parse word of the form  $00v$  and are mutually crooked, then they have a parse word of the form  $01v00$ .*
- *If  $T_1$  and  $T_2$  have no parse word of the form  $00v$ , then the only possibilities for the 2-tuple*

*(level of leaf 2 in  $T_1$ , level of leaf  $n - 1$  in  $T_2$ )*

*are  $(2, 3)$  and  $(k, 2)$  for some  $k \geq 2$ .*

*Moreover, if  $T_1$  and  $T_2$  are weakly mutually crooked, the only possibilities*

are  $(2, 3)$  and  $(k, 2)$  for some  $2 \leq k \leq 4$ .  
 Moreover, if  $T_1$  and  $T_2$  are mutually crooked, the only possibilities are  $(2, 3)$   
 and  $(k, 2)$  for some  $2 \leq k \leq 3$ .

Finally, we give an interesting conjecture that has been explicitly verified for  $n \leq 12$ . (The statement does not hold for general binary trees.)

**Conjecture 22.** *Let  $n \geq 4$ . Every pair of  $n$ -leaf path trees parses a word of the form  $u00v$  for some (possibly empty)  $u, v$ .*

#### REFERENCES

- [1] K. Appel and W. Haken, Every planar map is four colorable I: discharging, Illinois J. Math. 21 (1977) 429–490.
- [2] K. Appel, W. Haken, J. Koch, Every planar map is four colorable II: reducibility, Illinois J. Math. 21 (1977) 491–567.
- [3] S. A. Csar, R. Sengupta, W. Suksompong, On a subposet of the Tamari lattice, available from <http://arxiv.org/abs/1108.5690>.
- [4] L. Kauffman, Map coloring and the vector cross product, J. Combin. Theory Ser. B 48 (1990) 145–154.
- [5] E. Rowland, PARSEWORDS, available from <http://www.cs.uwaterloo.ca/~erowland/packages.html>.
- [6] D. Zeilberger, LOU, available from <http://www.math.rutgers.edu/~zeilberg/programs.html>.

SCHOOL OF MATHEMATICS, 206 CHURCH ST. S.E., MINNEAPOLIS, MN 55455, USA

SCHOOL OF COMPUTER SCIENCE, UNIVERSITY OF WATERLOO, WATERLOO, ON N2L 3G1,  
 CANADA

DEPARTMENT OF MATHEMATICS, RUTGERS UNIVERSITY, PISCATAWAY, NJ 08854, USA