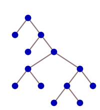
TOWARD A LANGUAGE THEORETIC PROOF OF THE FOUR COLOR THEOREM

BOBBE J. COOPER, ERIC S. ROWLAND, AND DORON ZEILBERGER

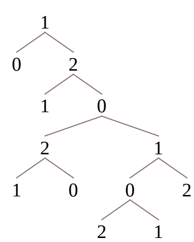
Abstract. This paper considers the problem of showing that every pair of binary trees with the same number of leaves parses a common word under a certain simple grammar. We enumerate the common parse words for several infinite families of tree pairs and discuss several ways to reduce the problem of finding 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 $0 \to 12$, $0 \to 21$, $1 \to 02$, $1 \to 20$, $2 \to 01$, $2 \to 10$. 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

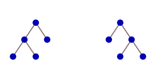


parses the word 0110212, as can be seen by this labeling:



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

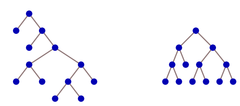
Certainly G is *ambiguous* — there exist distinct trees that parse the same word; for example, the trees



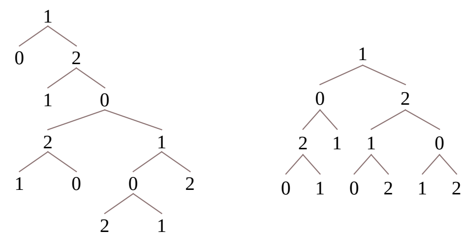
Date: June 5, 2010.

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

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 [3] proved this theorem (in a slightly different form) 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 and Haken [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. Section 2 determines explicit common parse words for several simple parameterized families of tree pairs. In Section 3 we establish existence of parse words for more general families. In Section 4 we enumerate the common parse words of a 3-parameter family of tree pairs. We conclude in Section 5 by discussing in more generality methods of reducing the problem of finding a common parse word for a pair of trees.

A *Mathematica* package [4] and a Maple package [5] 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. Parameterized families

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 (i,j,k) of $(0,1,2)$,*

$$|w|_i \equiv |w|_j \not\equiv |w|_k \equiv |w| \mod 2.$$

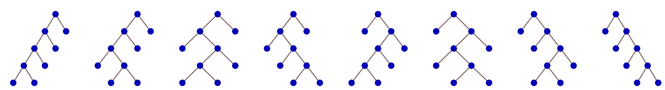
Moreover, the root of T receives the label k when parsing w.

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. □

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

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.

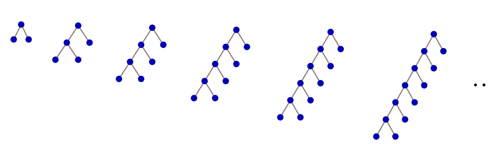
Because of their linear structure, path trees are simpler to work with than binary trees in general, so the emphasis of this paper is on path trees. Indeed, several infinite families of pairs of path trees can be shown to satisfy Theorem 1 directly and have only a few parse words. We take up this task in the current section. 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.

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^k1v. 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 $\text{ParseWords}(T_1,T_2)\neq\{\}$.

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

Theorem 3. ParseWords(LeftCombTree(n), RightCombTree(n)) =

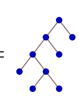
$$\begin{cases} \{01^{n-2}2\} & \textit{if } n \geq 2 \textit{ is even} \\ \{01^{n-2}0\} & \textit{if } n \geq 3 \textit{ is odd.} \end{cases}$$

Proof. We build a common parse word from left to right — up LeftCombTree(n) and down 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 RightCombTree(n) receives the label 1, the non-leaf (internal) vertex on the second level of RightCombTree(n) receives 2, and therefore the internal vertex on the third level of 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. □

Note from the proof of this theorem that the internal labels corresponding to a common parse word of LeftCombTree(n) and 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 LeftTurnTree(m, n) be the $(m+n)$-leaf path tree corresponding to $l^m r^{n-2}$, and let 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,

LeftTurnTree(2, 3) =  .

The following theorem is a special case of the general treatment of two turn trees given in Section 4.

Theorem 4. *For* $m \geq 1$,

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

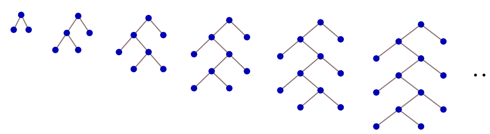
Proof. Without loss of generality, label the last leaf of each tree 0. The roots of the trees receive the same label, and thus the respective parents of the last leaf of each tree must receive 1 and 2 in some order, and the first leaf must be labeled 0. This implies that the last m leaves are labeled 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 LeftTurnTree(m, n)), namely 12, 10, and 01. Each of the first two options can been 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 RightTurnTree$(1, m+n-1)$ is labeled 0, which contradicts leaf $n+1$ receiving 0. □

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

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

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

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



Let RightCrookedTree(n) be the path tree corresponding to $(rl)^{(n-2)/2}$ — the left–right reflection of 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 $\mathrm{mod}(n, 3)$ be the smallest nonnegative integer congruent to n modulo 3.

Theorem 5. ParseWords(LeftCombTree(n), RightCrookedTree(n)) =

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

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

Inductively, assume that LeftCombTree$(n-1)$ and 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 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 RightCrookedTree$(n-1)$ receives 0, and for odd $n-1$ the left bottom leaf of 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  at the bottom of RightCrookedTree$(n-1)$ to form RightCrookedTree(n) and insert  at the corresponding place in LeftCombTree$(n-1)$ to form LeftCombTree(n). Label the new bottom leaves of RightCrookedTree(n) 12 if $n-1$ is even and 21 if $n-1$ is odd; we can label the corresponding leaves of 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 $0 \to 2, 1 \to 0, 2 \to 1$ 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 $n-1$. □

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

Theorem 6. ParseWords(LeftCombTree(n), LeftCrookedTree(n)) =

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

Theorem 7. *For $n \geq 2$,*

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

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 LeftCrookedTree($n-2$) and RightCrookedTree($n-2$) by

respectively to obtain LeftCrookedTree(n) and RightCrookedTree(n). Because the three new leaves are leaves $(n-1)/2$, $(n+1)/2$, and $(n+3)/2$ in both LeftCrookedTree(n) and 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

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

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 LeftCrookedTree(n) and 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 ∴ 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 LeftCrookedTree(n) and 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 $1 \leq i \leq \frac{n}{2} - 1$. □

3. 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 4. Note that, where stated, these results apply to not just path trees but binary trees in general.

Proposition 8. *Let $n \geq 3$. If the ith 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 7 consisting of LeftCrookedTree(n) and 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 kth 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. □

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

Proposition 9. *Suppose T_1' and T_2' are n-leaf binary trees with a common parse word. Extend each tree by attaching $\wedge$ 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. □

In the next proposition we consider extending a tree T by inserting $\wedge$ 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 $\wedge$ to the sibling of leaf i, and then reattach S to a leaf of the new $\wedge$ as follows. If leaf i is a left leaf, attach S to the right leaf of the new $\wedge$; 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.

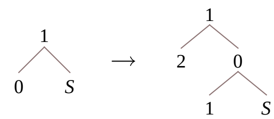
Proposition 10. *Suppose T_1' and T_2' are n-leaf binary trees with a common parse word. Extend T_1' by attaching $\wedge$ to leaf i, obtaining T_1. Extend T_2' to obtain T_2 by duplicating leaf i. Then*

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

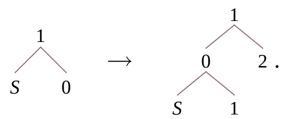
In particular, T_1 and T_2 have a common parse word.

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

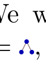


Clearly T_1 parses both of these words, so we have found a parse word for the pair. Moreover, every parse word of T_1 and T_2 arises uniquely in this way. $\square$

In Section 2 we referred to the two leaves of maximal depth in 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 11. *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 LeftCombTree(2) = , 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 9 gives twice as many parse words for T and LeftCombTree(n) as parse words for T' and LeftCombTree($n-1$). If $i > 1$, then leaf i is a right leaf in LeftCombTree($n-1$), so Proposition 10 gives the same number of parse words as for T' and LeftCombTree($n-1$). $\square$

Theorem 12. *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 ith 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 8. 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 ith leaf of T_2 is not the right bottom leaf.

Therefore we may assume that the ith 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 ith leaf of T_2 is the left bottom leaf, then we can apply Proposition 9 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 10 applies. □

4. A pair of turn trees

In this section we give three theorems that collectively determine the number of parse words of $\mathrm{LeftTurnTree}(m,n)$ and $\mathrm{RightTurnTree}(k,m+n-k)$. Note that by Theorem 12 the number of parse words is nonzero.

Theorem 13. *For $m \geq 1$, $k \geq 1$, and $\max(2, k-m+2) \leq n \leq k$,*

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

Proof. The bottom leaves of $\mathrm{LeftTurnTree}(m,n)$ (which are leaves $n-1$ and n) correspond to leaves which are on consecutive levels in $\mathrm{RightTurnTree}(k, m+n-k)$, so we can apply Proposition 10 to see that

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

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 $\mathrm{LeftTurnTree}(m,2) = \mathrm{LeftCombTree}(m+2)$ and $\mathrm{RightTurnTree}(k-(n-2), m+n-k)$, which have a unique parse word class by Theorem 11. □

Let

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

By considering the left–right reflections of these two trees, we see that $a(m,k) = a(k,m)$. Theorem 14 determines the number of parse words of $\mathrm{LeftTurnTree}(m,n)$ and $\mathrm{RightTurnTree}(k, m+n-k)$ for $n \geq k+2$ in terms of $a(m,k)$, and Theorem 16 evaluates $a(m,k)$.

Theorem 14. *For $m \geq 1$, $k \geq 1$, and $n \geq k+2$,*

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

Proof. If $n > k+2$, then the bottom leaves of $\mathrm{LeftTurnTree}(m,n)$ correspond to leaves which are on consecutive levels in $\mathrm{RightTurnTree}(k, m+n-k)$, so we can apply Proposition 10 to see that

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

If our hypothesis applies to this new, smaller tree pair, we may continue reducing in exactly the same way until we reach $\mathrm{LeftTurnTree}(m,k+2)$ and $\mathrm{RightTurnTree}(k,m+2)$. Leaves k and $k+1$ are bottom leaves in both these trees, so by Proposition 9 we have

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

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 15. *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 $\operatorname{mod}(i, 3)$. Then

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

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

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

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

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

Similarly,

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

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

Theorem 16. *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

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

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 LeftTurnTree$(m,k+1)$ and RightTurnTree$(k,m+1)$ with L and R produces a parse word for the pair. Such pairs (L,R) are in bijection to 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 LeftTurnTree$(m,k+1)$ and 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 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 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

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

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

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

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 LeftTurnTree$(m,k+1)$ and a child of a j leaf in 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 15, 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

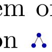
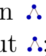
$$(M-2)(M-1)(M+1)\, a(m,k) = 0.$$

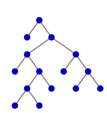
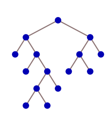
where M is the forward shift operator in the variable m.

When $m = 1$ or $k = 1$ one of the two trees is a comb tree, and by Theorem 11 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 $k = 1$. □

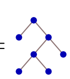
5. Reducing a pair of trees

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.

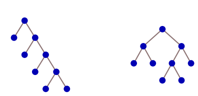
5.1. **Decomposable pairs.** Recall that if T_1 and T_2 are n-leaf trees such that leaves i and $i+1$ are siblings in both trees, then Proposition 9 reduces the problem of finding a parse word for T_1 and T_2 to the problem of finding a parse word for the pair of $(n-1)$-leaf trees in which the common  has been removed. Our first observation is that there is nothing special about ; 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

$T_1 =$  $T_2 =$ 

share the branch system

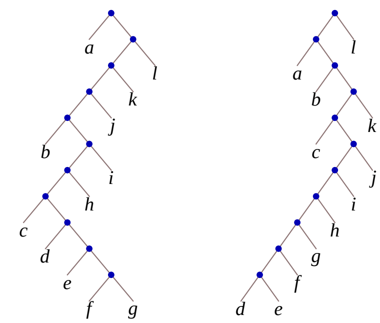
$S =$ 

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

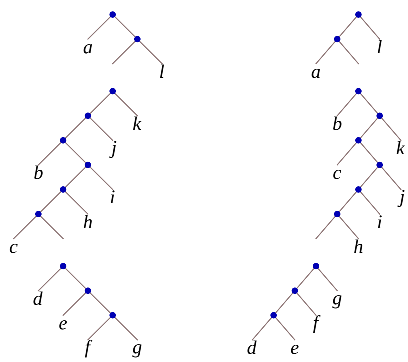


Given a common parse word $w_1 w_2 w_3 w_4 w_5$ of this pair of 5-leaf trees, we can find a common parse word of the original pair of 8-leaf trees by taking any valid labeling of S and permuting the alphabet so that the root receives the label w_2.

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



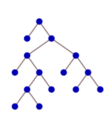
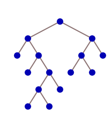
(where corresponding leaves have been given the same label). Breaking the trees at levels 2 and 8 as



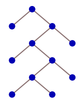
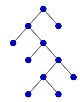
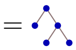
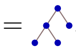
produces the same partition $\{\{a,l\},\{b,c,h,i,j,k\},\{d,e,f,g\}\}$ of the leaves in both trees. Thus, to find a parse word for a original pair it suffices to find parse words for the subtree pairs. Proposition 2 guarantees that we can reattach the subtrees consistently, since every binary tree that 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 3 consisting of LeftCombTree(n) and RightCombTree(n) is indecomposable, as is the pair in Theorem 5 consisting of LeftCombTree(n) and RightCrookedTree(n). On the other hand, breaking the trees LeftCombTree(n) and 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 6.

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

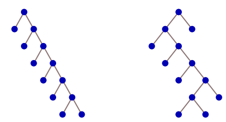
$T_1 =$  $T_2 =$ 

can also be decomposed into the two pairs

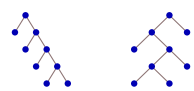
$T_1' =$  $T_2' =$ , $S_1 =$  $S_2 =$ .

5.2. **Pairs that are not mutually crooked.** We showed in Proposition 10 that if leaves i and $i+1$ are siblings in T_1 and are on consecutive levels in T_2 then this pair of trees is reducible. It is natural then to ask whether a tree pair in which leaves i and $i+1$ are on consecutive levels in both trees is reducible.

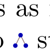
First let us consider the pair



which has the common parse word 0001220. The three consecutive 0s label leaves in both trees that are arranged in a right comb structure, and shortening each comb by two leaves produces the pair



which parses 01220. In other words, we obtain a parse word for the larger pair by simply inserting two 0s corresponding to the two added leaves. Let us see why this works. In the first tree, the 4-leaf right comb subtree whose first three leaves are labeled 0 has internal vertex labels 1, 2, and 1; in the second tree, the corresponding 4-leaf right comb subtree has internal vertex labels 2, 1, and 2. In both cases, the internal vertex labels alternate between 1 and 2, so shortening the comb by two leaves preserves the label of the root and the labels of the bottom leaves. Hence we still have a valid labeling.

The situation will be the same even if the two combs have different orientations, and even if the trees are not path trees but binary trees in general. We formalize this as follows. The relevant extending operation is *triplicating* leaf i — inserting two  structures in T' so as to obtain three left leaves in T if leaf i is a left leaf in T' and three right leaves in T if leaf i is a right leaf in T', analogous to duplicating a leaf as discussed in Section 3.

Theorem 17. *Let T_1' and T_2' be $(n-2)$-leaf binary trees. Let $1 \le i \le n-2$, and let T_1 and T_2 be the n-leaf trees obtained from T_1' and T_2' by triplicating leaf i. If $w = w_1 \cdots w_{n-2}$ is a parse word for T_1' and T_2', then $w_1 \cdots w_{i-1} w_i w_i w_i w_{i+1} \cdots w_{n-2}$ is a parse word for T_1 and T_2.*

A pair of n-leaf trees T_1 and T_2 is *weakly mutually crooked* if it cannot be obtained by triplicating some leaf i in a pair of $(n-2)$-leaf trees. To prove that every pair of binary trees has a parse word, by the previous theorem it suffices to consider pairs of weakly mutually crooked trees.

However, it appears that something stronger is true. A pair of n-leaf trees T_1 and T_2 is *mutually crooked* if it cannot be obtained by duplicating some leaf i in a pair of $(n-1)$-leaf trees; that is, there is no i such that leaves i and $i+1$ are on consecutive levels in both T_1 and T_2. Experimental evidence suggests that in fact it suffices to consider pairs of mutually crooked trees.

Conjecture 18. *Let T_1' and T_2' be $(n-1)$-leaf binary trees. Let $1 \le i \le n-1$, 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 above 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 7) are mutually crooked, but for $n \ge 5$ 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 3 and 4 provide additional examples of pairs that fail to be mutually crooked.

5.3. **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.

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

Conjecture 19. *Let $n \ge 4$, 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.*

- *T_1 and T_2 have no parse word of the form $10v00$.*
- *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*

  $$(\textit{level of leaf } 2 \textit{ in } T_1, \textit{level of leaf } n-1 \textit{ in } T_2)$$

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

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

Conjecture 20. *Let $n \ge 4$. Every pair of n-leaf path trees parses a word of the form $u00v$ for some (possibly empty) u, v.*

References

[1] Kenneth Appel and Wolfgang Haken, Every planar map is four colorable I: discharging, *Illinois Journal of Mathematics* **21** (1977) 429–490.
[2] Kenneth Appel, Wolfgang Haken, and John Koch, Every planar map is four colorable II: reducibility, *Illinois Journal of Mathematics* **21** (1977) 491–567.
[3] Louis Kauffman, Map coloring and the vector cross product, *Journal of Combinatorial Theory, Series B* **48** (1990) 145–154.
[4] Eric Rowland, ParseWords, available from `http://www.math.tulane.edu/~erowland/packages.html`.
[5] Doron Zeilberger, LOU, available from `http://www.math.rutgers.edu/~zeilberg/programs.html`.

School of Mathematics, 206 Church St. S.E., Minneapolis, MN 55455, USA

Department of Mathematics, Tulane University, New Orleans, LA 70118, USA

Department of Mathematics, Rutgers University, Piscataway, NJ 08854, USA