Séminaire Lotharingien de Combinatoire **86B** (2022)
Article #45, 12 pp.

Proceedings of the 34th Conference on Formal Power Series and Algebraic Combinatorics (Bangalore)

Planar Tanglegram Layouts and Single Edge Insertion

Kevin Liu*[1]

[1]*Department of Mathematics, University of Washington, Seattle, WA 98125, USA*

Abstract. Tanglegrams are formed by taking two rooted binary trees T and S with the same number of leaves and uniquely matching each leaf in T with a leaf in S. They are usually represented using layouts that embed the trees and matching in the plane. Planar tanglegrams are tanglegrams that have a layout with no crossings. Previous work on planar tanglegrams include enumeration, a Tanglegram Kuratowski Theorem, and an algorithm for drawing a planar layout. In this paper, we characterize all planar layouts of a planar tanglegram. We then apply our work to inserting a single edge into a planar tanglegram.

Keywords: tanglegram, tree, layout, crossing, planar, edge insertion

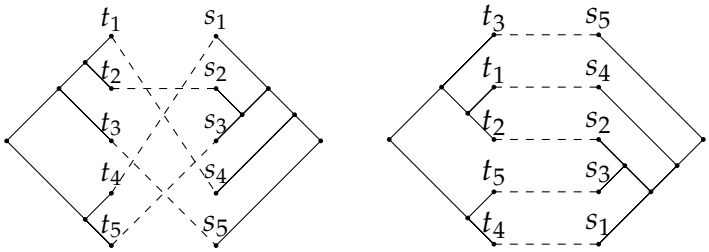

Figure 1: Two layouts for the same tanglegram.

1 Introduction

Let T and S be two rooted binary trees with leaves labeled as $\{t_i\}_{i\in I}$ and $\{s_j\}_{j\in J}$, respectively, where $I, J \subseteq \mathbb{N}$ are finite index sets of the same size. If we let $\phi\colon I \to J$ be a bijection, then we can denote a tanglegram as (T, S, ϕ), where ϕ indicates that t_i is matched with $s_{\phi(i)}$. A *layout* of a tanglegram draws T, S, and the edges $(t_i, s_{\phi(i)})$ in the plane such that T is planarly embedded left of the line $x = 0$ with all leaves on $x = 0$, S is planarly embedded right of the line $x = 1$ with all leaves on $x = 1$, and the edges $(t_i, s_{\phi(i)})$ are drawn using straight lines. Examples are shown in Figure 1. A *crossing* is any pair of edges $(t_i, s_{\phi(i)})$ and $(t_j, s_{\phi(j)})$ that intersect in the layout, and the *crossing number* of a tanglegram (T, S, ϕ), denoted $\mathrm{crt}(T, S, \phi)$, is the minimum number of crossings

*kliu15@uw.edu. Partially supported by the National Science Foundation grant DMS-1764012.

over all layouts of (T, S, ϕ). The Tanglegram Layout Problem attempts to efficiently find a layout that achieves the crossing number.

Tanglegrams initially arose in biology and computer science. Biologists use binary trees to model evolution and tanglegrams to model relationships between species. Finding optimal layouts helps determine how two species may have co-evolved [15]. Applications in computer science include clustering, decomposition of programs into layers, or analyzing the difference in hierarchy between similar programs or different versions of the same program [3]. Combinatorial interest in tanglegrams developed more recently. Matsen *et al.* [15] formalized tanglegrams as mathematical objects and described connections with phylogenetics. Billey, Konvalinka, and Matsen [2] enumerated tanglegrams and constructed an algorithm to generate them uniformly at random. Subsequently, Ralaivaosaona, Ravelomanana, and Wagner counted planar tanglegrams [17], Gessel counted several variations of tanglegrams using combinatorial species [8], and Konvalinka and Wagner studied the properties of random tanglegrams [12].

The Tanglegram Layout Problem has connections to the problem of drawing a graph with the fewest number of crossings possible. The crossing number of a graph G, denoted $\mathrm{cr}(G)$, is the minimum number of crossings over all drawings of G. Determining if $\mathrm{cr}(G) \leq k$ for some $k \in \mathbb{N}$ is NP-complete [7], and the same is true for determining if $\mathrm{crt}(T, S, \phi) \leq k$ [6]. Known results in graph drawings sometimes have analogous results in tanglegram layouts, and some have approached the Tanglegram Layout Problem by translating known results about graphs to tanglegrams. Czabarka, Székely, and Wagner recently constructed a Tanglegram Kuratowski Theorem characterizing planar tanglegrams. Previously, Lozano *et al.* constructed an algorithm for drawing a planar layout [14]. Anderson *et al.* recently translated results in graph drawings to prove that removing a between-tree edge $(t_i, s_{\phi(i)})$ from a tanglegram reduces crossing number by at most $n - 3$, and they produce a family of tanglegrams to show that this bound is sharp [1].

Given the difficulty of minimizing crossings in graph drawings, some have studied approximating $\mathrm{cr}(G)$ rather than finding it exactly. One approach to this is edge insertion. The Edge Insertion Problem for graphs starts with a graph G and an edge $e \in G$ such that $G \setminus \{e\}$ is planar, and attempts to find a drawing of G in the plane so that the drawing of $G \setminus \{e\}$ is planar and the number of crossings from e is minimized. This problem is well studied. A linear-time algorithm exists to solve it, and some bounds have been found relating an optimal drawing of G and a solution to the Edge Insertion Problem for $G \setminus \{e\}$ with $\{e\}$ inserted [9, 10]. The Edge Insertion Problem generalizes to the Multiple Edge Insertion Problem, where we insert several edges $\{e_1, \ldots, e_n\}$ into planar $G \setminus \{e_1, \ldots, e_n\}$ optimally, and current approximation algorithms for $\mathrm{cr}(G)$ still use multiple edge insertion with planar subgraphs [4]. Given the role that edge insertion with planar subgraphs plays in graph drawing, it is plausible that edge insertion can play a similar role for tanglegram layouts.

In this paper, we outline several results involving planar tanglegram layouts and single edge insertion. Full details for these results and their proofs can be found in [13]. We start by characterizing the planar layouts of a planar tanglegram. For a planar tanglegram (T, S, ϕ), we will define a *leaf-matched pair* (u, v) as a pair of internal vertices $u \in T$ and $v \in S$ whose descendant leaves are matched by ϕ, and we will define an operation called a *paired flip*. By adding steps to the `Untangle` algorithm by Lozano *et al.* for drawing a planar layout of a planar tanglegram, we construct `ModifiedUntangle` (Algorithm 1), which also identifies leaf-matched pairs and stores them in a set L, obtaining the following result.

Theorem 1.1. *Let (T, S, ϕ) be a planar tanglegram, and let $\mathscr{P}(T, S, \phi)$ denote its collection of planar layouts. Let the output of `ModifiedUntangle`(T, S, ϕ) be the layout (X, Y) and set of leaf-matched pairs L. Every $(X', Y') \in \mathscr{P}(T, S, \phi)$ can be obtained by starting with (X, Y) and performing a sequence of paired flips at $(u, v) \in L$.*

Note that when T and S have more than one vertex, the roots of T and S always form a leaf-matched pair of (T, S, ϕ). If this is the only leaf-matched pair, then we call the tanglegram *irreducible*. Letting $\text{size}(T, S, \phi)$ be the common number of leaves in T and S, the generating function

$$H(x) = \sum_{\text{irreducible planar } (T,S,\phi)} x^{\text{size}(T,S,\phi)} \tag{1.1}$$

was studied in [17], where the authors used the convention that the coefficient of x^2 is $\frac{1}{2}$ to obtain their results enumerating planar tanglegrams. We maintain this convention. If

$$F(x, q) = \sum_{\text{planar } (T,S,\phi)} x^{\text{size}(T,S,\phi)} q^{|\{\text{leaf-matched pairs of } (T,S,\phi)\}|}, \tag{1.2}$$

then the coefficient of $x^n q^k$ counts the number of planar tanglegrams of size n with k leaf-matched pairs. The following result allows us to find these coefficients.

Theorem 1.2. *The generating function $F(x, q)$ satisfies the relation*

$$F(x, q) = q \cdot H(F(x, q)) + x + \frac{q \cdot F(x^2, q^2)}{2}.$$

Finally, we consider the tanglegram analogue of (single) edge insertion, which we state below. Using our previous results on leaf-matched pairs, we will describe our `Insertion` algorithm for efficiently solving this problem.

Problem 1.3 (Tanglegram Single Edge Insertion)**.** *Given a tanglegram (T, S, ϕ) and a planar subtanglegram $(T_I, S_{\phi(I)}, \phi|_I)$ induced by the index set $I = [n] \setminus \{i\}$ for $i \in [n]$, find a layout of (T, S, ϕ) that restricts to a planar layout of $(T_I, S_{\phi(I)}, \phi|_I)$ and has the minimal number of crossings possible.*

Theorem 1.4. *The `Insertion` Algorithm solves the Tanglegram Single Edge Insertion Problem in $O(n^2)$ time and space, where n is the size of the tanglegram.*

2 Preliminaries

A *rooted binary tree* T is a tree in which every vertex has either zero or two children, and where a designated vertex called the root, denoted $\text{root}(T)$, is allowed to have degree 2. A vertex that has children is called an *internal vertex*, and a vertex with no children is called a *leaf*. If v has children v_1 and v_2, we call v the parent of v_1 and v_2. We say that a vertex v_1 is a *descendant* of v_k or v_k is an *ancestor* of v_1 if there is a sequence of vertices $v_1, v_2, \ldots, v_k$ such that v_{i+1} is the parent of v_i for $i = 1, 2, \ldots, k-1$, and we use the notation $v_1 < v_k$ or $v_k > v_1$ to denote this. When needed, we will use a subscript with the name of a tree to indicate ancestry in specific trees, such as $v_k >_T v_1$.

For an internal vertex $v \in T$, the *subtree rooted at* v is the tree formed by all vertices u with $u \leq v$, and this subtree then has v as its root. Using subtrees, we can represent trees using the nested lists notation from Section 2.3.2 of [11]. Each set of parenthesis represents a subtree, and the label for the root of this subtree is written to the left of the parenthesis. Unless otherwise stated, we will index leaves with $[n] = \{1, 2, \ldots, n\}$, and usually we will omit labels for internal vertices.

All trees are considered up to isomorphism, so relabeling vertices does not produce a different tree. Given an internal vertex $v \in T$, a *flip* at vertex v is the operation that interchanges the order of the children for all $u \leq v$. Pictorially, if we start with a drawing of a tree T, a flip at $v \in T$ reflects the subtree rooted at v, which motivates the name "flip." Note that for any tree T, flips generate all trees isomorphic to T.

Tanglegrams (T, S, ϕ) are formed from a pair of rooted binary trees T, S and a bijection ϕ matching their leaves. The *size* of the tanglegram (T, S, ϕ) is the common number of leaves in T or S. We will call the edges in T and S *tree edges* and call the edges induced by ϕ *between-tree edges*. For any vertex $u \in T$ or $v \in S$, we will use $\text{Lf}(u)$ and $\text{Lf}(v)$ to respectively denote the set of leaves that are descendants of u in T and the descendants of v in S. As with trees, we consider tanglegrams up to isomorphism. Since flips generate all trees isomorphic to the underlying trees, they also generate all tanglegrams isomorphic to a given tanglegram.

Our notation for tanglegram layouts builds on the notation used in [14]. In any layout, the number of crossings is completely determined by the order of the leaves in the two trees and the bijection ϕ matching these leaves, as the between-tree edges $(t_i, s_{\phi(i)})$ and $(t_j, s_{\phi(j)})$ intersect when t_i is embedded above t_j and $s_{\phi(i)}$ is embedded below $s_{\phi(j)}$. Since we are primarily interested in the number of crossings rather than specific coordinates of the plane embedding, we give the following definition.

Definition 2.1. Let (T, S, ϕ) be a tanglegram drawn in the plane with a given layout. The *leaf order* of the given layout is a pair of ordered lists (X, Y), where X and Y respectively list the leaves of T and S in order of appearance from top to bottom in the layout.

One can view the leaf order of a layout (X, Y) as an equivalence class of layouts, where two layouts are equivalent if they draw the leaves of T and S in the same order

from top to bottom. To recover a layout from the ordered lists (X, Y), one can draw the leaves listed in X and Y from top to bottom respectively on $x = 0$ and $x = 1$, and then use the information from T, S, and ϕ to draw the trees and between-tree edges. Flips generate all trees isomorphic to T or S, so they can act on leaf orders (X, Y) to obtain all possible leaf orders, where a flip at an internal vertex u acts on (X, Y) by reversing the order of the elements in $\mathrm{Lf}(u)$ in the appropriate list X or Y. Throughout this paper, we abuse terminology and refer to this pair of lists (X, Y) also as a tanglegram layout.

Finally, we will define induced subtrees and induced subtanglegrams using a similar definition as in [5]. Notice that layouts (X', Y') of subtanglegrams correspond to taking sub-lists in layouts (X, Y) of the original tanglegram.

Definition 2.2. Let T be a tree with leaves indexed by $[n]$. For any $I \subseteq [n]$, the *rooted binary subtree induced by* I, denoted T_I, is formed by taking the minimal subtree of T containing the leaves indexed by I and suppressing all internal vertices that have only one child. For a tanglegram (T, S, ϕ) with leaves indexed by $[n]$, the *subtanglegram induced by* I is the tanglegram $(T_I, S_{\phi(I)}, \phi|_I)$, that is, the tanglegram formed from the induced subtrees T_I and $S_{\phi(I)}$ with leaves matched using the restriction $\phi|_I$.

Example 2.3. Consider the tanglegram (T, S, ϕ) shown on the left in Figure 2 with

$$T = (((t_1, t_2), t_3), (t_4, t_5)) \qquad S = (((s_1, (s_2, s_3)), s_4), s_5)$$

i	1	2	3	4	5
$\phi(i)$	4	2	5	1	3

By writing the leaves of T and S in the order given, we obtain the layout $(X, Y) = (t_1t_2t_3t_4t_5, s_1s_2s_3s_4s_5)$. A flip at the root of $((s_1, (s_2, s_3)), s_4)$ yields $(t_1t_2t_3t_4t_5, s_4s_3s_2s_1s_5)$. Taking sub-lists of $(t_1t_2t_3t_4t_5, s_4s_3s_2s_1s_5)$ based on the elements in the sets $\{1, 2, 4, 5\}$ and $\phi(\{1, 2, 4, 5\})$, we obtain the layout $(t_1t_2t_4t_5, s_4s_3s_2s_1)$ for $(T_{\{1,2,4,5\}}, S_{\phi(\{1,2,4,5\})}, \phi|_{\{1,2,4,5\}})$. Using T and S, we produce the drawings in Figure 2 corresponding to these sequences.

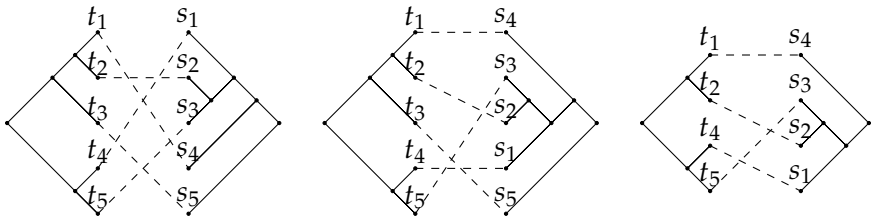

Figure 2: Layouts from Example 2.3.

3 Leaf-matched pairs and planar tanglegram layouts

In this section, we discuss leaf-matched pairs, which allow us to characterize planar tanglegram layouts. We start with some definitions, followed by an example in Figure 3. We then give our `ModifiedUntangle` algorithm.

Definition 3.1. Let (T, S, ϕ) be a tanglegram with layout (X, Y). A pair of internal vertices (u, v) with $u \in T$ and $v \in S$ is a *leaf-matched pair* of (T, S, ϕ) if $\mathrm{Lf}(u)$ and $\mathrm{Lf}(v)$ are matched by ϕ. A *paired flip* at (u, v) is the operation on (T, S, ϕ) corresponding to a flip at u and a flip at v. This maps (X, Y) to the layout (X', Y'), where X' is the image of X after a flip at u and Y' is the image of Y after a flip at v.

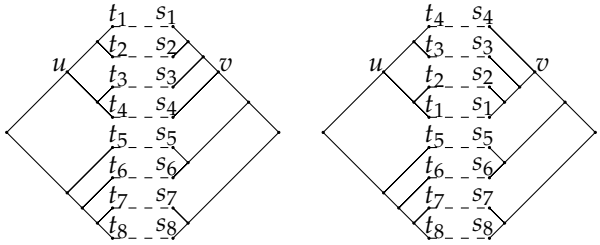

Figure 3: Starting with the layout on the left, a paired flip at the leaf-matched pair (u, v) results in the layout on the right.

Algorithm 1: `ModifiedUntangle` (based on [14, Algorithm 2])

Input: planar tanglegram (T, S, ϕ) with leaves $\{t_1, \ldots, t_n\}$ and $\{s_1, \ldots, s_n\}$
Output: a planar layout (X, Y) of (T, S, ϕ), list of leaf-matched pairs $L \subseteq T \times S$

$P :=$ Boolean table with $P[u, v] = \mathit{False}$ for all vertices $u \in T$, $v \in S$
set $P[t_i, s_{\phi(i)}] = \mathit{True}$ for all $i \in [n]$
recursively set $P[u, v] = \mathit{True}$ for internal vertices $u \in T, v \in S$ if there exists $u' \leq_T u$, $v' \leq_S v$ with $P[u', v'] = \mathit{True}$
$X :=$ (root(T)), $Y :=$ (root(S)) as ordered lists
$E := \{(\mathrm{root}(T), \mathrm{root}(S))\}$ as a set of edges
$L := \emptyset$
while $X \cup Y$ *contains an internal vertex of* T *or* S **do**
    $u :=$ an internal vertex of T or S with highest degree in the bipartite graph $G = (X, Y, E)$
    if $u \in X$ **then**
        if u *has degree 1 in* G **then**
            update $L := L \cup (u, v)$, where v is the unique neighbor of u in G
        update $X, E :=$ `Refine`(X, Y, u, E, P)
    else if $u \in Y$ **then**
        if u *has degree 1 in* G **then**
            update $L := L \cup (v, u)$, where v is the unique neighbor of u in G
        update $Y, E :=$ `Refine`(Y, X, u, E, P)
return $(X, Y), L$

Algorithm 2: `Refine` (based on [14, Algorithm 3])

Input: ordered lists of vertices (A, B), $u \in A$, edges E on $A \cup B$, Boolean table P
Output: A, E after u has been replaced with its children

$u_1, u_2 :=$ children of u in $T \cup S$
foreach $j \in [m]$ *such that* $(u, b_j) \in E$ *where* $B = (b_1, \ldots, b_m)$ **do**
  update $E := E \setminus \{(u, b_j)\}$ `// replace edges in` E `involving` u `with those involving` u_1, u_2
  foreach $i \in \{1, 2\}$ **do**
    if $P[u_i, b_j] = True$ **then**
      update $E := E \cup \{(u_i, b_j)\}$
$k := \max\{j \in [m] : (u_1, b_j) \in E\}$ `// last vertex in` B `adjacent to` u_1
if $j > k$ *for all* $(u_2, b_j) \in E$ **then**
  replace u with $u_1 u_2$ in A
else
  replace u with $u_2 u_1$ in A
return A, E

Theorem 3.2. *[14] For any planar tanglegram (T, S, ϕ) of size n, `ModifiedUntangle` terminates in a planar layout (X, Y) and runs in $O(n^2)$ time and space.*

Lemma 3.3. *If (T, S, ϕ) is a planar tanglegram, then the set L returned by `ModifiedUntangle` is the set of leaf-matched pairs of (T, S, ϕ).*

Using the techniques from the proof of Theorem 3.2 and Lemma 3.3, we can establish Theorem 1.1. Hence, we can use the output of `ModifiedUntangle` to find all planar layouts of a tanglegram. Theorem 1.1 also gives us an interesting corollary about the flip graph of a planar tanglegram, which we now define.

Definition 3.4. Let (T, S, ϕ) be a planar tanglegram. Define the *flip graph of* (T, S, ϕ) as $\Gamma(T, S, \phi) = (V, E)$ with vertices $v_{(X,Y)} \in V$ corresponding to planar layouts (X, Y), and edges $(v_{(X,Y)}, v_{(X',Y')}) \in E$ if (X', Y') can be obtained from (X, Y) by a paired flip at some leaf-matched pair (u, v) of (T, S, ϕ).

Corollary 3.5. *The flip graph of a planar tanglegram is connected.*

We can count the number of tanglegrams of size n with k leaf-matched pairs. Recall the generating functions $H(x)$ and $F(x, q)$ defined in (1.1) and (1.2). Generalizing the arguments in [17, Theorem 1] establishes Theorem 1.2. Using this result, it takes a computer-algebra system a moment to generate several coefficients of $F(x, q)$. The coefficient of $x^n q^k$ counts the number of planar tanglegrams of size n with k leaf-matched pairs, and we collect these coefficients in Table 1. See [16, A349409] for more terms. The corresponding planar tanglegrams for $n = 4$ are shown in Figure 4.

n,k	1	2	3	4	5	6	7	total
2	1							1
3	1	1						2
4	5	4	2					11
5	34	28	11	3				76
6	273	239	102	29	6			649
7	2436	2283	1045	325	73	11		6173
8	23391	23475	11539	3852	968	181	23	63429

Table 1: The number of planar tanglegrams of size n with k leaf-matched pairs.

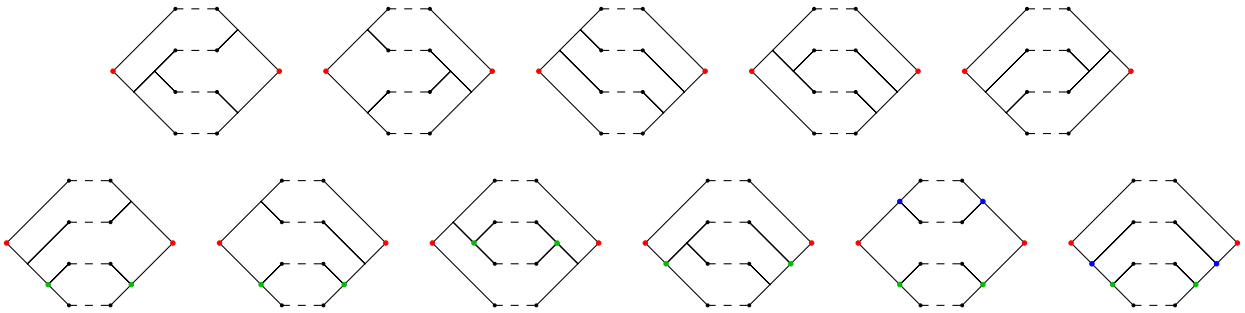

Figure 4: The 11 planar tanglegrams of size 4 with color-coded leaf-matched pairs. The first five tanglegrams have one leaf-matched pair, and hence are irreducible. The following four have two pairs, and the final two have three pairs.

4 The Tanglegram Single Edge Insertion Problem

In this section, we describe our `Insertion` Algorithm for solving the Tanglegram Single Edge Insertion Problem. Throughout this section, fix a tanglegram (T, S, ϕ) of size n, fix $I = [n] \setminus \{i\}$ for some $i \in [n]$, and let (X, Y) be a layout of (T, S, ϕ) that restricts to a planar layout of $(T_I, S_{\phi(I)}, \phi|_I)$. We use the notation

$$\begin{aligned} u_0 &= \text{parent of } t_i \in T, \\ v_0 &= \text{parent of } s_{\phi(i)} \in S, \\ L(I) &= \{\text{leaf-matched pairs of } (T_I, S_{\phi(I)}, \phi|_I)\}, \\ L(I)_T &= \{(u,v) \in L(I) : u >_T t_i, v \not>_S s_{\phi(i)}\}, \\ L(I)_S &= \{(u,v) \in L(I) : u \not>_T t_i, v >_S s_{\phi(i)}\}. \end{aligned} \tag{4.1}$$

An example of the sets $L(I)_T$ and $L(I)_S$ is given in Figure 5. Note that u_0 and v_0 are also the unique internal vertices in (T, S, ϕ) that are not in $(T_I, S_{\phi(I)}, \phi|_I)$. Using these sets with our results from Section 3, the Tanglegram Single Edge Insertion Problem reduces to certain operations at the elements above.

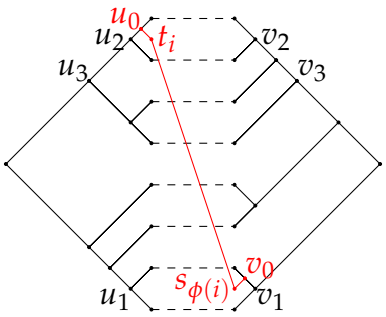

Figure 5: For the tanglegram given above with subtanglegram $(T_I, S_{\phi(I)}, \phi|_I)$ shown in black, $L(I)_T = \{(u_2, v_2), (u_3, v_3)\}$ and $L(I)_S = \{(u_1, v_1)\}$. Note that $(\text{root}(T), \text{root}(S)) \in L(I)$ is not an element of either set since $\text{root}(T) >_T t_i$ and $\text{root}(S) >_S s_{\phi(i)}$.

Definition 4.1. Let T be a tree, let $u \in T$ be an internal vertex, and let u_1, u_2 be the children of u. A *subtree switch* at u is the operation on T that interchanges the two subtrees rooted at u_1 and u_2, while maintaining the relative order of all leaves within each subtree.

Observe that a subtree switch at u is equivalent to a flip at u and a flip at each of its children. On a layout (X, Y) of a tanglegram (T, S, ϕ), the action of a subtree switch at u interchanges the order of the two sublists corresponding to $\text{Lf}(u_1)$ and $\text{Lf}(u_2)$.

Lemma 4.2. *A solution to the Tanglegram Single Edge Insertion Problem can be obtained by starting at (X, Y) and performing a sequence of paired flips at $(u, v) \in L(I)_T \cup L(I)_S$ and subtree switches at u_0 and v_0.*

Identifying a solution in Lemma 4.2 will require some cases. Notice that $u >_T u_0$ for any $(u, v) \in L(I)_T$ and $v >_S v_0$ for any $(u, v) \in L(I)_S$, as u_0 and v_0 are respectively the parents of t_i and $s_{\phi(i)}$. Additionally, whenever $L(I)_T$ and $L(I)_S$ are nonempty, each set has a unique "maximal" element, which we denote

$$\begin{aligned}(u_{\text{Tmax}}, v_{\text{Tmax}}) &= \text{unique } (u, v) \in L(I)_T \text{ such that } u \geq_T u' \text{ for all } (u', v') \in L(I)_T, \\ (u_{\text{Smax}}, v_{\text{Smax}}) &= \text{unique } (u, v) \in L(I)_S \text{ such that } v \geq_S v' \text{ for all } (u', v') \in L(I)_S.\end{aligned} \tag{4.2}$$

Lemma 4.3. *If $L(I)_S \neq \emptyset$ and $u_0 >_T u_{\text{Smax}}$, then $L(I)_T = \emptyset$. If $L(I)_T \neq \emptyset$ and $v_0 >_S v_{\text{Tmax}}$, then $L(I)_S = \emptyset$.*

When $u_0 >_T u_{\text{Smax}}$, the previous lemma implies $L(I)_T = \emptyset$. In this case, we find a solution in Lemma 4.2 by considering operations at $u_0, (u, v) \in L(I)_S$, and v_0 in the unique order such that no element is considered before all of its ancestors have been considered. We make paired flip and subtree switch choices greedily based on the crossings that can be affected by operations at the given vertices, but not by operations we consider afterwards. An example is shown in Figure 6. The case $v_0 >_S v_{\text{Tmax}}$ is similar.

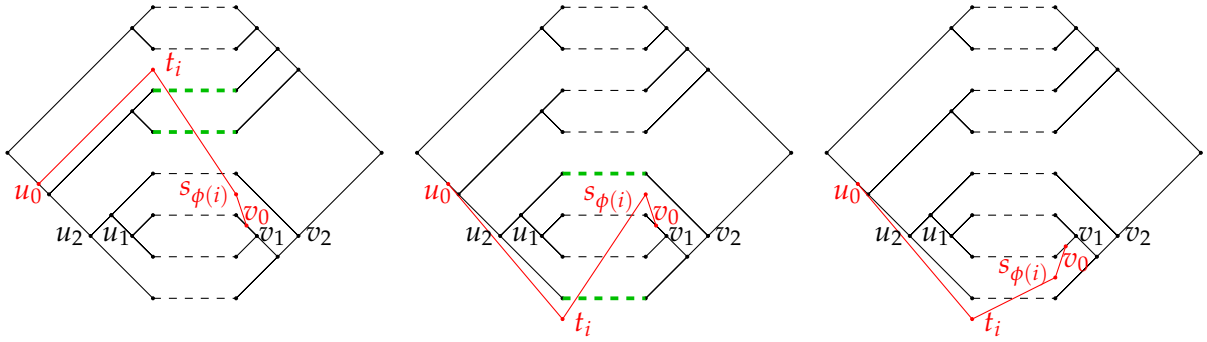

Figure 6: In this example, $L(I)_S = \{(u_1, v_1), (u_2, v_2)\}$ with $(u_{\text{Smax}}, v_{\text{Smax}}) = (u_2, v_2)$. Starting with (X, Y) on the left, a subtree switch at u_0 determines whether $(t_i, s_{\phi(i)})$ crosses the edges in green. Performing a subtree switch at u_0 removes these crossings, resulting in the middle layout. Subsequent steps include no operation at (u_2, v_2), a paired flip at (u_1, v_1), and no operation at v_0, returning the layout on the right.

For all remaining cases, we first consider the elements of $L(I)_T$ in descending order with respect to ancestry, followed by the elements of $L(I)_S$ in descending order. We again make paired flip choices greedily based on crossings that cannot be affected by operations we consider afterwards. We then consider the four layouts generated by all combinations of subtree switches at u_0 and v_0, and we return the layout with the fewest number of crossings. An example of this procedure is shown in Figure 7.

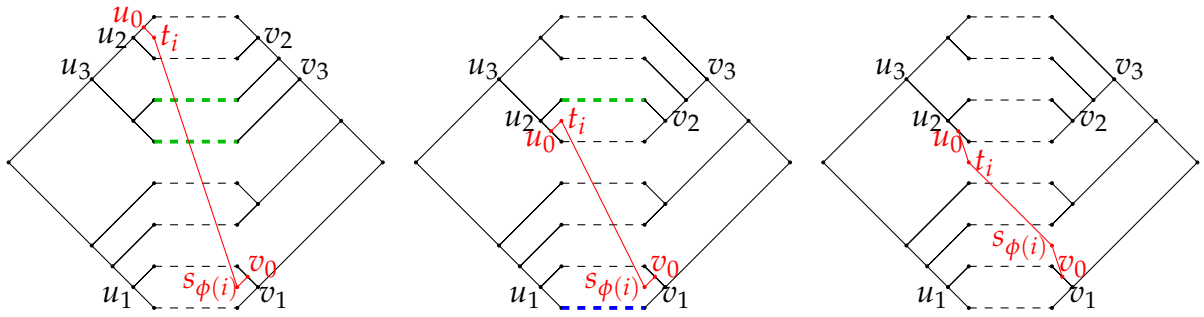

Figure 7: Starting with (X, Y) on the left, we perform a paired flip at (u_3, v_3) to minimize crossings between $(t_i, s_{\phi(i)})$ and the edges in green, obtaining the layout in the middle. For (u_2, v_2) and (u_1, v_1), we respectively consider the edges shown in green and blue, and ultimately do not perform either of those paired flips. After considering all choices of subtree switches at u_0 and v_0, we return the layout shown on the right.

For our `Insertion` Algorithm, first use `ModifiedUntangle` on $(T, S, \phi|_I)$ to find a layout (X, Y) of (T, S, ϕ) that restricts to a planar layout of $(T_I, S_{\phi(I)}, \phi|_I)$. Then construct $L(I)_T$ and $L(I)_S$. Afterwards, check for the different cases involving u_{Smax} or v_{Tmax}, and proceed accordingly. After formalizing this procedure, we obtain Theorem 1.4.

5 Future Work

In Section 3, we defined the flip graph of a planar tanglegram. While paired flips will generate all vertices in this graph, it is possible that some flips do not produce a new layout, as tanglegrams are considered up to isomorphism on T and S. One simple example of this is the unique tanglegram of size 2, where a paired flip at the roots of both trees does not produce a new layout. As such, we pose the following problem.

Problem 5.1. *For any planar tanglegram (T, S, ϕ), characterize the flip graph $\Gamma(T, S, \phi) = (V, E)$. In particular, determine $|V|$, $|E|$, and $\deg(v)$ for $v \in V$.*

Our next problem involves generating tanglegrams uniformly at random. Billey, Konvalinka, and Matsen previously constructed an algorithm that generates tanglegrams uniformly at random in [2], and we propose a corresponding problem for the planar case.

Problem 5.2. *Find an efficient algorithm for generating a planar tanglegram of size n uniformly at random.*

Finally, one can find examples where a solution to the Tanglegram Single Edge Insertion Problem is not a solution to the Tanglegram Layout Problem, such as the ones constructed in [13, Corollary 4.18]. However, we could consider flips that do not preserve planarity of the subtanglegram but do reduce the number of crossings.

Problem 5.3. *Can the Insertion algorithm be modified to create an efficient approximation algorithm for the crossing number of an almost planar tanglegram?*

Acknowledgements

We would like to thank Sara Billey for suggesting tanglegrams as an area of research and for valuable feedback. We would also like to thank Matjaž Konvalinka, Stark Ryan, and the anonymous reviewers for valuable feedback on earlier versions of this paper.

References

[1] R. Anderson et al. "Analogies between the crossing number and the tangle crossing number". *Electron. J. Combin.* **25**.4 (2018), Paper No. 4.24, 15.

[2] S. C. Billey, M. Konvalinka, and F. A. Matsen. "On the enumeration of tanglegrams and tangled chains". *J. Combin. Theory Ser. A* **146** (2017), pp. 239–263.

[3] K. Buchin, M. Buchin, J. Byrka, M. Nöllenburg, Y. Okamoto, R. I. Silveira, and A. Wolff. "Drawing (complete) binary tanglegrams: hardness, approximation, fixed-parameter tractability". *Algorithmica* **62**.1-2 (2012), pp. 309–332.

[4] J. Chuzhoy, S. Mahabadi, and Z. Tan. "Towards better approximation of graph crossing number". *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)* (2020), pp. 73–84.

[5] E. Czabarka, L. A. Székely, and S. Wagner. "A tanglegram Kuratowski theorem". *J. Graph Theory* **90**.2 (2019), pp. 111–122.

[6] H. Fernau, M. Kaufmann, and M. Poths. "Comparing trees via crossing minimization". *J. Comput. Syst. Sci.* **76** (2010), pp. 593–608.

[7] M. R. Garey and D. S. Johnson. "Crossing number is NP-complete". *SIAM J. Algebraic Discrete Methods* **4**.3 (1983), pp. 312–316.

[8] I. M. Gessel. "Counting tanglegrams with species". *J. Comb. Theory, Ser. A* **184** (2021), p. 105498.

[9] C. Gutwenger, P. Mutzel, and R. Weiskircher. "Inserting an edge into a planar graph." *Algorithmica* **41** (2001), pp. 289–308.

[10] P. Hliněný and G. Salazar. "On the Crossing Number of Almost Planar Graphs". *Graph Drawing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 162–173.

[11] D. E. Knuth. *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. USA: Addison Wesley Longman Publishing Co., Inc., 1997.

[12] M. Konvalinka and S. Wagner. "The shape of random tanglegrams". *Adv. in Appl. Math.* **78** (2016), pp. 76–93.

[13] K. Liu. "Characterizing planar tanglegram layouts and applications to edge insertion problems". 2022. arXiv:2201.10533.

[14] A. Lozano, R. Pinter, O. Rokhlenko, G. Valiente, and M. Ziv-Ukelson. "Seeded Tree Alignment and Planar Tanglegram Layout". *WABI*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 98–110.

[15] F. A. Matsen, S. C. Billey, A. Kas, and M. Konvalinka. "Tanglegrams: a reduction tool for mathematical phylogenetics". *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **15**.1 (2015), pp. 343–349.

[16] OEIS Foundation Inc. "The On-Line Encyclopedia of Integer Sequences". http://oeis.org. 2021.

[17] D. Ralaivaosaona, J. B. Ravelomanana, and S. Wagner. "Counting Planar Tanglegrams". *29th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2018)*. Vol. 110. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2018, 32:1–32:18.