The Power of Local Optimization: Approximation Algorithms for Maximum-leaf Spanning Tree (DRAFT)*

Hsueh-I Lu and R. Ravi

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-96-05
January 1996

The Power of Local Optimization: Approximation Algorithms for Maximum-leaf Spanning Tree (DRAFT)*

Hsueh-I Lu
hil@cs.brown.edu
Department of Computer Science
Brown University

R. Ravi
ravi+@andrew.cmu.edu
Graduate School of Industrial Administration
Carnegie Mellon University

November 1995

Abstract

Given an undirected graph G, finding a spanning tree of G with maximum number of leaves is NP-complete. We use the simple technique of local optimization to provide the first approximation algorithms for this problem. Our algorithms run in polynomial time to produce locally optimal solutions. We prove that locally optimal solutions to this problem are globally near-optimal. In particular, we prove that two such algorithms have performance ratios of 5 and 3. The latter algorithm employs more powerful local-improvement steps than the former and hence has higher running time. This may indicate an interesting trade-off between the performance ratios and the running times of the series of algorithms we describe.

Keywords: Approximation algorithms, NP-complete problems, Performance ratio, Local optimization, Communication network design, Combinatorial algorithms.

1 Introduction

Given an undirected graph $G = (V, E)$, the Maximum Leaf Spanning Tree problem is to find a spanning tree of G with maximum number of leaves. This problem finds applications in communication networks, circuit layouts and in other graph-theoretic problems [25]. An interesting application is mentioned in [22]: In [4], E. W. Dijkstra studied the problem of self-stabilizing a set of processors in the presence of distributed control and proposed a solution based on mutual exclusion. A variant of this solution [26] seeks a spanning tree of a graph such that the product of the degrees of all the nodes in the tree is minimum. We can use a spanning tree with maximum number of leaves as a heuristic solution (with no guarantees) for the latter problem.

Maximum Leaf Spanning Tree is NP-complete [9]. In this paper we use the simple technique of local optimization to provide the first approximation algorithms for this problem. These algorithms perform a series of local-improvement steps until a local optimum is reached and output a locally optimal solution. This notion of applying local-improvement heuristics to hard optimization problems was around 1958 [3], long before the invention of NP-completeness [14]. It has been applied heuristically to solve a variety of NP-hard problems in combinatorial optimization [20]. A breakthrough in this area was the work of Fürer and Raghavachari [6], who applied this technique

*The preliminary version of this paper appeared in *Proceedings, Thirtieth Annual Allerton Conference on Communication, Control and Computing* (October 1992), pp. 533-542.

to the Minimum-Degree Spanning Tree problem. They showed a version of a local optimization algorithm that runs in polynomial time to produce near-optimal solutions. Our work is inspired by that of Fürer and Raghavachari. We demonstrate that local optimization can be used to design polynomial-time approximation algorithms for the Maximum Leaf Spanning Tree problem. Our algorithms output spanning trees with as many leaves as at least a constant fraction of the maximum.

Our algorithms perform local changes that increase the number of leaves in the resulting spanning tree. For the problem at hand, there is a natural notion of such a local change, namely, swapping a tree edge for a non-tree edge. This notion can be extended to include swapping many tree edges for an equal number of non-tree edges in a single local change operation. Our algorithms perform such local changes that increase the number of leaves until no improvement in the solution is possible and output a locally optimal solution as an approximate solution.

By varying the limit on the number of tree edges that can participate in a single local-change operation, we derive a series of approximation algorithms. Increasing this limit corresponds to allowing for more powerful local-improvement steps. Thus the higher is this limit, the higher is the running time of the corresponding algorithm. A local-change operation involving at most k tree edges is termed a *k-change*. Thus k-changes are costlier to implement than $(k-1)$-changes. Intuitively we expect that an algorithm using k-changes would perform better than an algorithm using only $(k-1)$-changes. In this paper, we corroborate this intuition and prove that local-improvement algorithms using 1-changes and 2-changes have performance ratios of 5 and 3, respectively. The following are our main results.

Theorem 1 There is an $O(n^4)$ algorithm using 1-changes for finding a spanning tree of any undirected graph G on n nodes such that the number of leaves of this spanning tree is at least as many as a fifth of the number of leaves of any spanning tree of G.

Theorem 2 There is an $O(n^7)$ algorithm using 2-changes for finding a spanning tree of any undirected graph G with n nodes such that the number of leaves of this spanning tree is at least as many as a third of the number of leaves of any spanning tree of G.

We prove that the running time of the approximation algorithm using k-changes increases with k as follows.

Lemma 1 The running time of the approximation algorithm using k-changes on a graph with n nodes is $O(n^{3k+1})$.

The pattern exhibited in Theorems 1 and 2 lead us to conjecture that there is a trade-off between the running times and the performance ratios exhibited in the series of algorithms we have defined.

Conjecture 1 The performance of the approximation algorithm using k-changes is strictly better than that of the algorithm using $(k-1)$-changes.

In the next section we survey related work. We then describe the approximation algorithms after introducing relevant definitions. In the following section, we illustrate our proof technique by providing a loose bound on the performance guarantee. In the subsequent sections we prove Theorems 1 and 2. We conclude with some remarks on related problems.

2 Related Work

The technique of local optimization has been applied to provide heuristic solutions for a variety of hard problems in combinatorial optimization. Chapter 19 of [20] surveys a few applications of this technique. Some notable examples are its applications to the graph-partitioning problem[15] and the TRAVELING SALESPERSON problems [17]. A number of complexity results relating to the paradigm of local optimality and the time complexity of computing a locally optimal solution are presented in [12, 19, 23, 27]. In this paper we are interested in the application of the local search techniques to design efficient approximation algorithms, namely, those that run in polynomial time and provide provably good solutions with values close to that of the optimum.

Fürer and Raghavachari [6] showed such an application of local search to the problem of computing a spanning tree of a given graph whose maximum degree is minimum. This is termed the MINIMUM DEGREE SPANNING TREE problem and is NP-complete [9]. Fürer and Raghavachari show that local optimization can be applied to produce spanning trees and even Steiner trees of nearly minimum degree [7]. The maximum degree of the spanning trees obtained by their polynomial-time approximation algorithms is at most one plus the maximum degree of the minimum-degree spanning trees. Unless P=NP, this performance guarantee is the best bound achievable in polynomial time. The result is generalized to work on the MINIMUM DEGREE STEINER TREE problem with the same performance guarantee [7]. In [24], Ravi, Raghavachari, and Klein generalized their technique to obtain approximation algorithms for a variety of minimum-degree-network-design problems. In this paper we add to the list of problems that are amenable to good approximate solutions by local-search methods. Very recently, using the technique of local optimization, Halldórsson [11] gives the approximation algorithms with the currently best performance ratios for a set of problems, including (a) k-SET COVER, (b) INDEPENDENT SET, VERTEX COVER, k-SET PACKING, k-DIMENSIONAL MATCHING and k-CLIQUE PACKING for $(k+1)$-claw free graphs, (c) INDUCED λ-COLORABLE SUBGRAPH and VERTEX ARBORICITY, which are special cases of HEREDITARY INDUCED SUBGRAPH, and etc.

Previous work on finding spanning trees with many leaves have focused on graphs with minimum degree at least k for some fixed $k \geq 3$. For such graphs, good lower bounds on the number of leaves achievable in a spanning tree are derived in [10, 16, 22, 25]. These lower bounds are typically proved algorithmically by constructing a spanning tree with the desired number of leaves. Thus these algorithms can be interpreted as approximation algorithms for the MAXIMUM LEAF SPANNING TREE problem for special classes of graphs in which the minimum degree of a node is at least k. However, to the best of our knowledge, no previous approximation algorithms for this problem are known in the general case that we consider in this paper. The best-known lower bound for the number of leaves achievable in a spanning tree of an n-node graph with minimum degree k is $(1 - b \ln k/k)n$ where b is any constant exceeding 2.5 and k is sufficiently large [16]. The best lower bounds for the cases $k = 3$ and 4 are $n/4$ and $2n/5$, respectively [16].

There has also been work on polynomial-time solutions to the problem of determining if a given graph has a spanning tree with at least k leaves for fixed k. The first such algorithm was due to Fellows and Langston [5]. The running time of their algorithm was improved by Bodlaender [2].

Papadimitriou and Yannakakis [21] identified a class of NP-hard optimization problems interreducible to one another using approximation-preserving reductions and thus took a step towards classifying NP-complete problems with respect to hardness of approximating them. They called this class MAX SNP. It has been shown that the problems complete for this class do not permit a fully polynomial-time approximation scheme unless P=NP [1]. Galbiati, Maffioli, and Morzenti show that MAXIMUM LEAF SPANNING TREE is MAX SNP-complete [8], which settles the open question raised in the preliminary version of this paper [18]. Therefore there exists some constant

$\epsilon_0 > 0$ such that there is no $(1+\epsilon_0)$-approximation for MAXIMUM LEAF SPANNING TREE unless P=NP. What the ϵ_0 is for the MAXIMUM LEAF SPANNING TREE problem remains open.

3 Preliminaries

Let G be an undirected graph. Let G' be a subgraph of G. We use $E(G')$ to denote the set of edges in G', and use $V(G')$ to denote the set of nodes in G'. A node v is *in* G' if $v \in V(G')$; an edge e is *in* G' if $e \in E(G')$. We shall sometimes refers to an edge uw of G as *the edge incident to* u *and* w. We say e is *incident to* G' if an endpoint of e is in G'. The *degree* of a node v in G' is the number of edges of G' that are incident to v. We use $V_i(G')$ to denote the set of nodes that have degree i in G, and use $\bar{V}_i(G')$ to denote the set of nodes that have degree at least i in G'. Clearly $\bar{V}_1(G) = V(G)$ if G' is connected and has at least two nodes.

Suppose G has more than one node. Let T be a spanning tree of G. A *leaf* of T is a node in $V_1(T)$. An *internal node* of T is a node in $\bar{V}_2(T)$. A *high-degree node* of T is a node in $\bar{V}_3(T)$. A leaf of T is *special* if it is adjacent in T to a high-degree node of T. A leaf of T is *normal* if it is not special. Therefore a normal leaf of T is adjacent in T to a node in $V_2(T)$ unless G has only two nodes. We use $V_{1S}(T)$ to denote the set of special leaves of T, and use $V_{1N}(T)$ to denote the set of normal leaves of T.

We use $\text{path}_T(u, w)$ to denote the path that connects u and w in T. Note that u and w are not necessarily distinct. We define $\text{path}_T(u, u)$ to be u itself. A *2-path* of T is a *maximal* path in T that contains only nodes in $V_2(T)$. Namely $\text{path}_T(u, w)$ is a 2-path of T if all nodes in $\text{path}_T(u, w)$ are in $V_2(T)$ and all nodes in $V_2(T) - V(\text{path}_T(u, w))$ are not adjacent to u or w in T. A 2-path P of T is *short* if $|V(P)| = 1$; otherwise it is *long*. We use $P_S(T)$ to denote the set of short 2-paths of T, and user $P_L(T)$ to denote the set of long 2-paths of T. We say a node v is *enclosed* by an edge uw in $G - T$ if $v \neq u$, $v \neq w$, and v is in $\text{path}_T(u, w)$.

A *region of* T is a connected component of the subgraph of T obtained from T by removing all nodes in $P_L(T)$ and all edges of T that have endpoints in $P_L(T)$. A region of T is *special* if it contains special leaves of T. A region of T is *normal* if it is not special. Suppose R and R' are two distinct regions of T. The long 2-path of T that connects R and R' in T is called the long 2-path *incident to* R *and* R'. The number of long 2-paths incident to a region of T is the *degree* of the region. A region is *internal* if it has degree at least two. A region is *external* if it is not internal. We use $R_{iS}(T)$ to denote the set of special regions of T that have degree i, and use $R_{iN}(T)$ to denote the set of normal regions of T that have degree i. Let $R_S(T)$ and $R_N(T)$ be the sets of special and normal regions of T, respectively.

There are some straightforward relations among these notations. Contracting every 2-path in a tree to a single edge and bounding the number of internal nodes (each resulting internal node has degree at least 3) in this contracted tree by the number of leaves, we have the following inequality.

$$|\bar{V}_3(T)| \leq |V_1(T)| - 2 \tag{1}$$

By discarding special leaves along with their incidental edges from a tree, contracting all 2-paths to single edges, and bounding the number of contracted edges in the resulting tree by the number of nodes, we have the following inequality.

$$|P_L(T)| + |P_S(T)| \leq |\bar{V}_3(T)| + |V_{1N}(T)| - 1 \tag{2}$$

Suppose we regard T as a tree of regions by contracting every long 2-path of T to a single edge. The number of nodes (i.e. regions of T) is exactly one plus the number of edges (i.e. long 2-paths

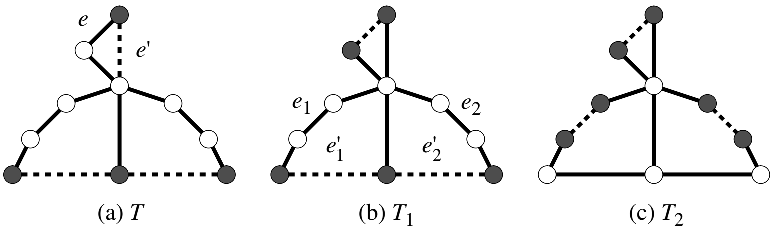


Figure 1: Examples of local improvements. In (a), the tree T is shown in dark lines and the non-tree edges are dashed. The 1-change $(\{e'\},\{e\})$ is a 1-improvement for T resulting in T_1 with one more leaf as shown in (b). T_1 does not admit any 1-improvement but admits the 2-improvement $(\{e'_1, e'_2\},\{e_1, e_2\})$ to give T_2 shown in (c).

of T). Also the sum of the degrees of nodes is exactly two times the number of edges.

$$
\begin{aligned}
|P_L(T)| &= |R_S(T)| + |R_N(T)| - 1 && (3)\\
2|P_L(T)| &= \sum_{i\geq 0} i \cdot (|R_{iS}(T)| + |R_{iN}(T)|)\,. && (4)
\end{aligned}
$$

Since each special region of T contains at least one special special leaf of T. Therefore we have

$$|R_S(T)| \leq |V_{1S}(T)|. \tag{5}$$

One can easily verify the following lemmas.

Lemma 2 Let T be a spanning tree of G. Let u, v and w be three distinct nodes of G. If v is in $\text{path}_T(u, w)$, then v is not a leaf of T.

4 The Algorithm

In this section we formally define k-changes and describe the approximation algorithms.

4.1 Edge Changes, Improvements, and Locally Optimal Trees

Suppose T is a spanning tree of G. Let E be a subset of edges in T and E' a subset of edges in $G-T$, where $|E| = |E'| \leq k$ for some $k \geq 1$. If $T + E' - E$ is still a spanning tree of G, then we call making edges in E' tree edges and making edges in E non-tree edges a *k-change*, denoted (E', E), for T. We use $T(E', E)$ to denote $T + E' - E$. A k-change (E', E) for T is a *k-improvement* if $|V_1(T(E', E))| > |V_1(T)|$.

A k-change can be regarded as a batched version of a sequence of k 1-changes, A k-change is more powerful than k 1-changes. In Figure 1 we illustrate a spanning tree T that admits a 1-improvement. On applying the 1-improvement we get a spanning tree T_1 that does not admit any 1-improvement but has a 2-improvement.

A *k-locally optimal tree* (k-LOT) of a given graph G is a spanning tree of G that does not admit any k-improvement. By definition an k-LOT is also a j-LOT for any $1 \leq j \leq k$. Since a spanning tree has exactly $n-1$ edges, a maximum-leaf spanning tree is just an $(n-1)$-LOT.

4.2 The Algorithm and Termination

The idea is simply using k-LOTs to approximate the maximum-leaf spanning trees for small values of k. The approximation algorithm is as follows.

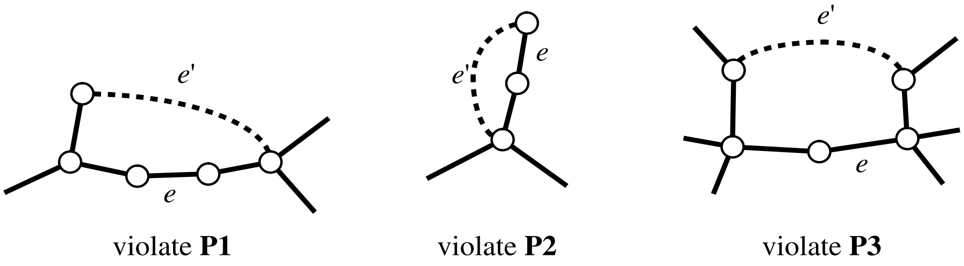


Figure 2: The dark edges represent tree edges. Each dashed edge is a non-tree edge violating one of the properties P1, P2, and P3. Note that the 1-change $(\{e'\}, \{e\})$ in each graph is a 1-improvement.

Procedure LocallyOptimalTree(G, k)
  Let T be a spanning tree of G.
  While there exists a k-improvement (E', E) for T do
    Let $T := T(E', E)$.
  Output T as a k-LOT of G.

The algorithm starts with an arbitrary spanning tree, which can be found in time $O(n^2)$, where n is the number of nodes in G. It then keeps applying k-improvements to the current spanning tree until it becomes a k-LOT. Clearly the second step dominates the time complexity. One can determine in polynomial time whether a tree is a k-LOT, since the number of distinct k-changes allowed for a spanning tree is at most $\binom{\binom{n}{2}-n+1}{k} \times \binom{n-1}{k}$, which is $O(n^{3k})$. The number of iterations required by the while-loop is at most n, since each iteration increases the number of leaves of the current T by at least one, and T has at most n leaves. Therefore it takes time $O(n^{3k+1})$ to obtain a k-LOT. This proves Lemma 1.

Our major concern is the *performance guarantee* of this approximation algorithm. In other words, how much less can the number of leaves of a k-LOT be than that of the optimum? In the following sections we prove that the performance ratios of 1-LOTs and 2-LOTs are 5 and 3, respectively. However, we conjecture that 2-LOTs have a better performance ratio than what we prove.

4.3 Properties of Locally Optimal Trees

Instead of jumping into proving the performance guarantees of 1-LOTs and 2-LOTs, we start with showing some straightforward properties of locally optimal trees. At the end of the section we illustrate our proof strategy by proving a loose bound on the performance ratio of 1-LOTs.

Let G be an undirected graph. Let T be a 1-LOT of G. By definition a 1-LOT has the following properties. They are illustrated in Figure 2.

P1 Let uw be an edge in $G - T$, where u is an internal node of T. Then $\text{path}_T(u, w)$ does not contain two consecutive nodes in $V_2(T)$.

P2 Let u be a normal leaf of T and let w be an internal node of T. Then there is no edge uw in $G - T$.

P3 Let uw be an edge in $G - T$, where u and w are both internal in T. Then $\text{path}_T(u, w)$ does not contain any node in $V_2(T)$.

The following is critical to showing a constant performance ratio of 1-LOTs.

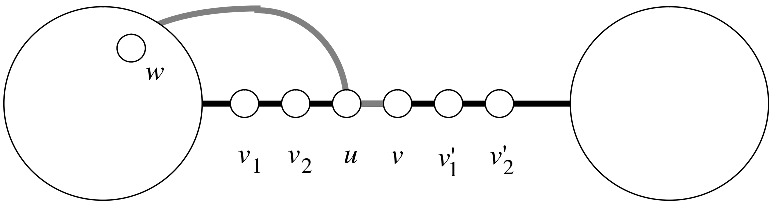


Figure 3:

Lemma 3 Let T be a 1-LOT of an undirected graph G. Let T' be a spanning tree of G. Then every 2-path of T contains at most four leaves of T'.

Proof Assume for a contradiction that a 2-path P of T contains more than four leaves of T'. Let v_1, v_2, v, v_1', and v_2' be five of them in order along P. Let v' be a node not in P. Let u be the node closest to v in $\text{path}_{T'}(v, v')$ such that u is not in P. Let w be the node closest to w in $\text{path}_{T'}(v, u)$. (See Figure 3.) Clearly uw is an edge of $T' - T$ and w is in P. By P3 we know that $T' - T$ does not have any edge incident to distinct nodes in P. It follows that w must be in $\text{path}_T(v_2, v_1')$. Therefore uw encloses either v_1 and v_2 or v_1' and v_2'. This contradicts the 1-optimality of T by P1. $\square$

With the help of the above lemma, we show that $|V_1(T')| \leq 10|V_1(T)|$ for any 1-LOT T of G and any spanning tree T' of G. Note that this relation holds even if T' is a maximum-leaf spanning tree of G.

$$\begin{aligned}
|V_1(T')| &= |V_1(T') \cap V_1(T)| + |V_1(T') \cap V_2(T)| + |V_1(T') \cap \bar{V}_3(T)| & \\
&\leq |V_1(T)| + |V_1(T') \cap V_2(T)| + |\bar{V}_3(T)| & \\
&\leq |V_1(T)| + 4(|P_S(T)| + |P_L(T)|) + |\bar{V}_3(T)| & \text{by Lemma 3} \\
&\leq |V_1(T)| + 4(|V_{1N}(T)| + |\bar{V}_3(T)|) + |\bar{V}_3(T)| & \text{by (2)} \\
&\leq 10|V_1(T)| & \text{by (1).}
\end{aligned}$$

5 Performance Guarantee of 1-LOTs

In this section we prove the following restatement of Theorem 1.

Theorem 3 Let T_0 be a 1-LOT of an undirected graph G_0. Let T_0 be a 1-LOT of G_0 and T_0' a spanning tree of G_0. Then $|V_1(T_0')| \leq 5|V_1(T_0)|$.

5.1 Augmentation

In order to simplify the proof of Theorem 3, we augment G_0 together with T_0 as follows. For every normal leaf of T_0 adjacent to a short 2-path, we subdivide the tree edge incident to the normal leaf by inserting a *pseudo-node* of degree 2 in this edge.

Lemma 4 Suppose G_0 is an directed graph and T_0 a 1-LOT of G_0. Let G and T be the augmented versions of G_0 and T_0, respectively. Then T is a 1-LOT of G.

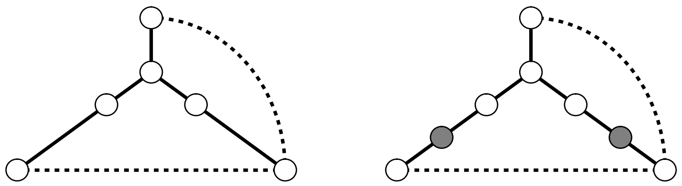


Figure 4:

Proof Assume for a contradiction that T is not a 1-LOT of G. It follows that there exists a 1-improvement for T involving addition of an edge $e \in G - T$. Since T_0 is a 1-LOT of G_0, a pseudo-node v must be enclosed by e in T. Note that no edge in $G - T$ is incident to v. To enclose v, e must be incident to the normal leaf of T_0 adjacent to v. Since T_0 is a 1-LOT and one of the endpoints of e is a normal leaf of T_0, by P2 the other endpoint of e should be a leaf of T_0 as well. By the definition of augmentation, e is incident to two leaves of T, too. However, no non-tree edge incident to two leaves can participate in a 1-improvement. This contradicts the fact that edge e yields a 1-improvement for T. Thus T is a 1-LOT. $\square$

An example is shown in Figure 4. The tree edges are in solid lines and the non-tree edges are in dashed lines. The tree at left is a 1-LOT with two external short 2-paths. The picture at right is the augmented graph and tree, where the gray nodes are the inserted pseudo-nodes. One can easily verify that the augmented tree is still a 1-LOT. Moreover this example shows that the augmentation might destroy the 2-optimality of a 2-LOT. It is not hard to see that the tree at left is a 2-LOT, and the tree at right is not a 2-LOT.

We say T is an *augmented 1-LOT of G* if T and G is augmented from T_0 and G_0, where T_0 is a 1-LOT of G_0. By definition of augmentation, an augmented 1-LOT does not have any external short 2-path. Therefore every normal of leaf of an augmented 1-LOT is a single-node external region. Therefore every augmented 1-LOT T has the following property.

$$|V_{1N}(T)| = |R_{1N}(T)| \leq |R_N(T)| \tag{6}$$

The following lemma ensures that if we can prove that the performance ratio of augmented 1-LOTs is at most five, then Theorem 3 is proved.

Lemma 5 Let T_0 be a 1-LOT of G_0. Let G and T be the augmented versions of G_0 and T_0, respectively. Then $|V_1(T)| = |V_1(T_0)| \leq |V_1(T_0^*)| \leq |V_1(T^*)|$ where T_0^* is a maximum-leaf spanning tree of G_0 and T^* is a maximum-leaf spanning tree of G.

Proof The equality is trivial since the augmentation does not change the number of leaves in T_0. The first inequality follows from the definition of T_0^*. The last inequality can be proved by constructing a spanning tree of G that has at least $|V_1(T_0^*)|$ leaves. We start with T_0^*. For each pseudo-node v to be inserted, if the unique edge whose subdivision resulted in v is in T_0^*, we subdivide this edge and insert the pseudo-node v in T_0^*. Otherwise, let the neighbors of v in G be v_1 and v_2. To augment T_0^* to include v, we simply attach v to v_1. In either case, we obtain a spanning tree of G that has at least $|V_1(T_0^*)|$ leaves. The lemma is thus proved. $\square$

It follows from the above lemma that

$$\frac{|V_1(T_0^*)|}{|V_1(T_0)|} \leq \frac{|V_1(T^*)|}{|V_1(T)|}.$$

Therefore in order to prove Theorem 3, it suffices to prove the following theorem.

Theorem 4 Let T be an augmented 1-LOT of G, where G is an undirected graph. Then $|V_1(T')| \leq 5|V_1(T)|$ holds for any spanning tree T' of G.

For the rest of the section we assume that T is an augmented 1-LOT of G, as given in Theorem 4.

5.2 Invalid Regions

Let T' be a spanning tree of G. A region R of T is *invalid in* T' if either (1) R is a special region of T and each special leaf of T in R is a leaf of T', or (2) R is an internal normal region of T. A region of T is *valid in* T' if it is not invalid in T'. Theorem 4 can be proved using the following lemma.

Lemma 6 Let T be an augmented 1-LOT of G. Let T' be a spanning tree of G. Then the number of leaves of T' that are in long 2-paths of T is at most

$$4|R_S(T)| - 2r + 3|V_{1N}(T)| - |V_{1N}(T) \cap V_1(T')|,$$

where r is the number of special regions of T that are invalid in T'.

Proof of Theorem 4 Since r is the number of special regions of T that are invalid in T', there are exactly $|R_S(T)| - r$ special regions of T that are valid in T'. By definition of invalid special regions, each special region of T that is valid in T' contains at least one special leaf of T that is not a leaf of T'. It follows that there are at least $|R_S(T)| - r$ special leaves of T that are not leaves of T'. Therefore

$$|V_1(T') \cap V_{1S}(T)| \leq |V_{1S}(T)| - (|R_S(T)| - r). \tag{7}$$

It follows that

$$\begin{aligned}
|V_1(T')| &= |V_1(T') \cap V_{1S}(T)| + |V_1(T') \cap V_{1N}(T)| + |V_1(T') \cap V(P_S(T))| + \\
&\quad |V_1(T') \cap V(P_L(T))| + |V_1(T') \cap \bar{V}_3(T)| \\
&\leq |V_1(T') \cap V_{1S}(T)| + |V_1(T') \cap V_{1N}(T)| + |P_S(T)| + \\
&\quad |V_1(T') \cap V(P_L(T))| + |\bar{V}_3(T)| \\
&\leq |V_1(T') \cap V_{1S}(T)| + |V_1(T') \cap V_{1N}(T)| + |P_S(T)| + \\
&\quad (4|R_S(T)| - r + 3|V_{1N}(T)| - |V_{1N}(T) \cap V_1(T')|) + |\bar{V}_3(T)| && \text{by Lemma 6} \\
&= |V_1(T') \cap V_{1S}(T)| + |P_S(T)| + \\
&\quad 4|R_S(T)| - r + 3|V_{1N}(T)| + |\bar{V}_3(T)| \\
&\leq |V_{1S}(T)| - (|R_S(T)| - r) + |P_S(T)| + \\
&\quad 4|R_S(T)| - r + 3|V_{1N}(T)| + |\bar{V}_3(T)| && \text{by (7)} \\
&= |V_{1S}(T)| + |P_S(T)| + 3|R_S(T)| + 3|V_{1N}(T)| + |\bar{V}_3(T)| \\
&\leq 3|V_{1S}(T)| + |P_S(T)| + |R_S(T)| + 3|V_{1N}(T)| + |\bar{V}_3(T)| && \text{by (5)} \\
&= |P_S(T)| + |R_S(T)| + 3|V_1(T)| + |\bar{V}_3(T)| \\
&\leq |P_S(T)| + |P_L(T)| - |R_N(T)| + 3|V_1(T)| + |\bar{V}_3(T)| && \text{by (3)} \\
&\leq |\bar{V}_3(T)| + |V_{1N}(T)| - |R_N(T)| + 3|V_1(T)| + |\bar{V}_3(T)| && \text{by (2)} \\
&\leq 2|\bar{V}_3(T)| + 3|V_1(T)| && \text{by (6)} \\
&\leq 5|V_1(T)| && \text{by (1).}
\end{aligned}$$

□

For the rest of the section we prove Lemma 6.

5.3 Charge of Regions and Deficient Long 2-paths

The key for proving Lemma 6 is a tighter bound on $|V_1(T') \cap V(P_L(T))|$ than the bound of $4|P_L(T)|$ given in Lemma 3. Let R be a region of T. Let the *charge of R in T'*, denoted $\text{charge}_{T'}(R)$, be defined by

$$\text{charge}_{T'}(R) = \begin{cases} 2 & \text{if } R \text{ is a special region of } T \\ 0 & \text{if } R \text{ is a normal leaf of } T \text{ and a leaf of } T' \\ 1 & \text{otherwise.} \end{cases}$$

We have the following refined version of Lemma 3.

Lemma 7 Let T be an augmented 1-LOT of G. Let T' be a spanning tree of G. Let R and R' be two distinct regions of T. Let P be the long 2-path of T incident to R and R'. Then P contains at most $\text{charge}_{T'}(R) + \text{charge}_{T'}(R')$ leaves of T'.

Let G, T, T', P, R, and R' be as in Lemma 7. The long 2-path P is *deficient in T'* if

- $\text{charge}_{T'}(R) + \text{charge}_{T'}(R') \geq 2$, and
- P contains at most $\text{charge}_{T'}(R) + \text{charge}_{T'}(R') - 2$ leaves of T'.

The following lemma relates the number of invalid regions and that of deficient long 2-paths.

Lemma 8 Let T be an augmented 1-LOT of G. Let T' be a spanning tree of G. Let k be the number of regions of T that are invalid in T'. Then there are at least $k-1$ distinct long 2-paths of T that are deficient in T'.

Lemma 6 can therefore be proved as follows.

Proof of Lemma 6 Let $d(R)$ be the degree of the region R of T. Then

$$\begin{aligned} \ell &= \sum_{R \in R_S(T) \cup R_N(T)} \text{charge}_{T'}(R) d(R) \\ k &= \text{the number of regions of } T \text{ that are invalid in } T'. \end{aligned}$$

It follows from Lemma 7 and Lemma 8 that the number of leaves of T' that are in the long 2-paths of T is at most $\ell - 2(k-1)$. We show that $\ell - 2(k-1)$ is at most $4|R_S(T)| - 2r + 3|V_{1N}(T)| - |V_{1N}(T) \cap V_1(T')|$.

Note that $\text{charge}_{T'}(R) = 0$ for every $R \in R_{1N}(T) \cap V_1(T')$. Therefore

$$\begin{aligned} \ell &= \sum_{R \in R_S(T)} \text{charge}_{T'}(R) d(R) + \sum_{R \in R_N(T) - (R_{1N}(T) \cap V_1(T'))} \text{charge}_{T'}(R) d(R) \\ &= \sum_{R \in R_S(T)} 2d(R) + \sum_{R \in R_N(T)} d(R) - \sum_{R \in R_{1N}(T) \cap V_1(T')} d(R) \\ &= 2\sum_{i \geq 1} i \cdot |R_{iS}(T)| + \sum_{i \geq 1} i \cdot |R_{iN}(T)| - |V_{1N}(T) \cap V_1(T')|. \end{aligned}$$

It follows from (4) and (3) that

$$\begin{aligned} \ell &\leq 4|P_L(T)| - \sum_{i \geq 1} i \cdot |R_{iN}(T)| - |V_{1N}(T) \cap V_1(T')| \\ &\leq 4|R_S(T)| + 4|R_N(T)| - 4 - \sum_{i \geq 1} i \cdot |R_{iN}(T)| - |V_{1N}(T) \cap V_1(T')|. \end{aligned} \tag{8}$$

By definition a region R of T is invalid in T' if (i) R is an internal normal region of T (i.e. $R \in R_N(T) - R_{1N}(T) - R_{0N}(T)$) or (ii) R is a special region of T that is in valid in T'. Therefore

$$\begin{aligned} 2(k-1) &= 2(|R_N(T)| - |R_{1N}(T)| - |R_{0N}(T)| + r - 1) \\ &\geq 2(|R_N(T)| - |R_{1N}(T)| + r - 2) \\ &= 2|R_N(T)| - 2|R_{1N}(T)| + 2r - 4, \end{aligned} \tag{9}$$

where the inequality above is by $|R_{0N}(T) \leq 1|$. Combining (8), (9), and (6), we have

$$\begin{aligned} \ell - 2(k-1) &\leq 4|R_S(T)| + 2|R_N(T)| - \sum_{i \geq 1} i \cdot |R_{iN}(T)| - |V_{1N}(T) \cap V_1(T')| + 2|R_{1N}(T)| - 2r \\ &\leq 4|R_S(T)| + |R_{1N}(T)| - |V_{1N}(T) \cap V_1(T')| + 2|R_{1N}(T)| - 2r \\ &= 4|R_S(T)| - 2r + 3|V_{1N}(T)| - |V_{1N}(T) \cap V_1(T')|. \end{aligned}$$

The lemma is proved. $\square$

We prove Lemma 7 and Lemma 8 respectively in the following two subsections.

5.4 Proof of Lemma 7

Let $i = \text{charge}_{T'}(R)$ and $j = \text{charge}_{T'}(R')$. Assume for a contradiction that P contains more than $i + j$ leaves of T'. Let $v_1, \ldots, v_i, v, v'_1, \ldots, v'_j$ be $i + j + 1$ of them in order along P from the side of R to that of R'.

1. $i = j = 0$
   Since $\text{charge}_{T'}(R) = \text{charge}_{T'}(R') = 0$, we know R and R' are both leaves of T' and normal leaves of T. It follows that T is a path. Since v is a leaf of T' but internal in T, we know there must be some edges in $T' - T$. By P2 and P3 we know that the only possible edge in $T' - T$ is the edge incident to R and R'. Since R and R' are both leaves of T', there is no way for v to be connected to R or R' in T'. This contradicts the fact that T' is a spanning tree of G.

2. $j \geq 1$
   Let v' be a node in R'. Let $P' = \text{path}_{T'}(v, v')$. Let u be the node closest to v in P' such that u is not in P. Let w be the node in $\text{path}_{T'}(u, v)$ such that uw is an edge in P'. Clearly uw is an edge of $T' - T$ and w is in P. It follows from Lemma 2 that none of $v_1, \ldots, v_i, v'_1, \ldots, v'_j$ is in P'. By P3 we know that there cannot be any edges in $T' - T$ that is incident to two distinct nodes in P. It follows that w must be in $\text{path}_T(v_j, v'_1)$.

   We first show that u is not in R'. Assume for a contradiction that u is in R'. Since w is in $\text{path}_T(v_j, v'_1)$, we know that uw encloses v'_1. By P1 and P2 we know that u must be a special leaf of T. Thus R' is a special region of T, which implies that $j = 2$. Since uw encloses v'_1 and v'_2, the 1-optimality of T is contradicted by P1.

   It can be shown that u is not in R. Assume for a contradiction that u is in R. For exactly the same reason as above, we know that $i \not\geq 1$. Therefore we have $i = 0$. It follows from the definition of $\text{charge}_{T'}(R)$ that R, which is exactly u, is a leaf of T'. This contradicts the fact that u is in P' by Lemma 2, since $u \neq v$ and $u \neq v'$.

   Clearly u is not in any region of T that is not R or R', since otherwise uw would enclose all the nodes of some long 2-path of T, and thus the 1-optimality of T is violated by P1. By definition of u we know that u is not in P.

It follows from the above discussion that u must be in a long 2-path of T that is not P. Therefore u and w are both internal nodes of T. This implies that $i = 0$, since otherwise uw must enclose either v_1 or v_1', and thus the 1-optimality of T is contradicted by P3. However $i = 0$ means that R is an external normal region which contains only a single normal leaf of T. Therefore uw is forced to enclose v_1'. Therefore the 1-optimality of T is contradicted by P3.

3. $i \geq 1$
   The contradiction for this case can be obtained in a way similar to case 2.

The lemma is thus proved. □

5.5 Proof of Lemma 8

If $k \leq 1$, the lemma trivially holds. Therefore we assume that $k \geq 2$. Let $R_1, \ldots, R_k$ be the k regions of T that are invalid in T'. We identify $k-1$ distinct long 2-paths, denoted $P_1, \ldots, P_{k-1}$, of T that are deficient in T'.

Let R be a region of T (not necessarily invalid in T'). Let $\mathrm{adj}(R)$ be the maximum subset of $V(P_L(T))$ such that each node in $\mathrm{adj}(R)$ is an endpoint of an edge of T that is incident to a node of R.

Claim 1 Let R be a region of T that is invalid in T'. Suppose u is a high-degree node in R, and w is a node not in $\mathrm{adj}(R) \cup R$. Then $\mathrm{path}_{T'}(u, w)$ contains an edge of T' that is incident to a node in $\mathrm{adj}(R)$ and a node not in $\mathrm{adj}(R) \cup R$.

Proof Let $P' = \mathrm{path}_{T'}(u, w)$. Clearly P' contains an edge $u'w'$ such that $u' \in \mathrm{adj}(R) \cup R$ and $w' \notin \mathrm{adj}(R) \cup R$. We show u' must be in $\mathrm{adj}(R)$.

By definition of invalid regions, R is either an internal normal region of T or a special region of T that is invalid in T'. Since T is augmented, R does not contain any normal leaf of T. By definition $\mathrm{adj}(R)$ contains only internal nodes of T. Therefore u' is not a normal leaf of T.

Since R is invalid in T', a special leaf of T in R, if any, must be a leaf of T'. It follows from Lemma 2 that P', i.e. $\mathrm{path}_{T'}(u, w)$, does not contain any special leaf of T in R, since u is not a leaf of T and w is not in R. Thus u' is not a special leaf of T. Therefore u' is internal in T, since u' is in P'.

Assume for a contradiction that u' is in R. It follows that $u'w'$ must enclose a node in $\mathrm{adj}(R)$. By P3 we know that w' is a leaf of T. Thus w' is in a region of T other than R. This contradicts the 1-optimality of T by P1, since $u'w'$ encloses all then nodes of some long 2-paths of T. We then know that u' is not in R. Therefore u' is in $\mathrm{adj}(R)$. □

By definition of invalid regions, it can be easily verified that each R_i contains at least one internal node of T. Let v_i be an arbitrary internal node of T in R_i for every $1 \leq i \leq k$. Let $P_i' = \mathrm{path}_{T'}(v_i, v_k)$, for every $1 \leq i \leq k-1$. Clearly v_k is not in $\mathrm{adj}(R_i) \cup R_i$ for every $1 \leq i \leq k-1$. By Claim 1 we know that P_i' contains an edge incident to a node in $\mathrm{adj}(R_i)$ and a node not in $\mathrm{adj}(R_i) \cup R_i$. Let $u_i w_i$, where $u_i \in \mathrm{adj}(R_i)$ and $w_i \notin \mathrm{adj}(R_i) \cup R_i$, be the edge in P_i' such that u_i is closest to v_i in P_i'. Let P_i be the long 2-path of T that contains u_i. We show by proving the following claim that $P_1, \ldots, P_{k-1}$ are $k-1$ distinct long 2-paths of T that are deficient in T'.

Claim 2 For every $1 \leq i \leq k-1$, P_i is deficient in T'. Moreover $P_j \neq P_i$ for every $1 \leq j \neq i \leq k-1$.

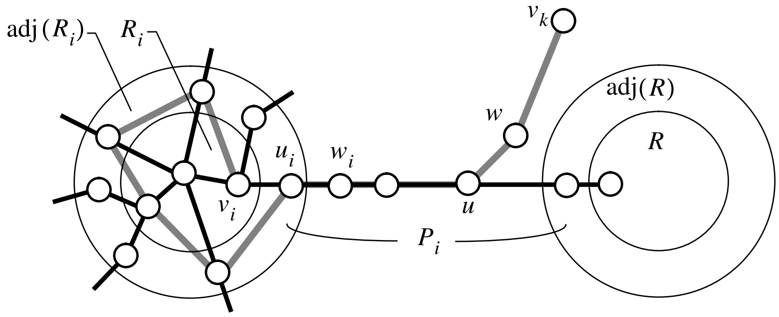


Figure 5: The dark lines are the edges of T. The gray path is P_i', i.e. $\text{path}_{T'}(v_i, v_k)$. We show in the proof that w must be in $\text{adj}(R) \cup R$.

Proof Since $u_i \neq v_i$, $u_i \neq v_k$, and u_i is in $\text{path}_{T'}(v_i, v_k)$, it follows from Lemma 2 that u_i is not a leaf of T'. Let R be the other region of T that P_i is incident to. Let w be the node closest to u_i in $\text{path}_{T'}(u_i, v_k)$ such that w is not in P_i. Let uw be the edge in $\text{path}_{T'}(u_i, w)$. Clearly u is in P_i, and thus each node in $\text{path}_T(u_i, u)$ are not leaves of T'. (See Figure 5).

We know that w cannot be in a region of T that is not R_i or R, since otherwise uw would enclose the nodes of some long 2-path of T, and thus the 1-optimality of T would be contradicted by P1.

We show that w is in $\text{adj}(R) \cup R$. Assume for a contradiction that w is not in $\text{adj}(R) \cup R$.

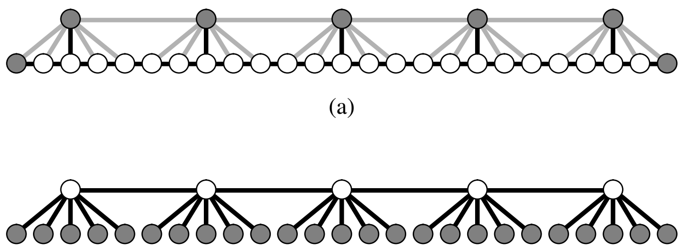
- $u = u_i$
  Clearly $w = w_i$. Since w is not in P_i by definition of w and w_i is not in $\text{adj}(R_i) \cup R_i$ by definition of w_i, we know that w is not in $\text{adj}(R_i) \cup R_i \cup P_i$. By assumption w is not in R. It follows that w must be in some long 2-path of T that is not P_i. Since w is not in $\text{adj}(R_i)$ by assumption and u is in $\text{adj}(R_i)$ by definition of u_i, we know that uw must enclose either a node in $\text{adj}(R_i)$ or a node in P_i. This contradicts the 1-optimality of T by P3, since u and w are both internal nodes of T.

- $u \neq u_i$
  Since $u \neq u_i$, we know that u is not in $\text{adj}(R_i) \cup R_i$. Note that w is not in $\text{adj}(R)$ by assumption. It follows that w cannot be in a long 2-path of T that is not P_i, since otherwise uw would enclose either a node in P_i or a node in $\text{adj}(R)$, which contradicts the 1-optimality of T by P3. Since w is not in P_i by definition of w, we know that w must be in some regions of T. However, we already know that w cannot be in any region of T that is not R_i or R. Therefore w must be in R_i by the assumption that w is not in $\text{adj}(R) \cup R$.

  Since w is in R_i, uw has to enclose u_i. By P1 and P2 we know that w is a special leaf of T, and thus R_i is a special region of T. Since R_i is invalid in T', w is a leaf of T'. This contradicts the fact that w is in $\text{path}_{T'}(v_i, v_k)$ by Lemma 2, since $w \neq v_i$ and $w \neq v_k$.

By definition of invalid regions we know that $\text{charge}_{T'}(R_i) \geq 1$. We prove the deficiency of P_i by showing that $\text{charge}_{T'}(R) \geq 1$ and the number of leaves of T' in P_i is at most $\text{charge}_{T'}(R) - 1$ for the following two cases.

- $w \in \text{adj}(R)$
  Since $w \notin P_i$, w must be in some long 2-path of T that is not P_i. By P3 we know that u is in $\text{adj}(R)$, since otherwise the node in $P_i \cap \text{adj}(R)$ would be enclosed by uw. Since u is in $\text{adj}(R)$ and u is in P_i, P_i does not contain any leaf of T'. It suffices to show that $\text{charge}_{T'}(R) \geq 1$.



(a)

(b)

Figure 6: (a) An example of 1-LOT. (b) The maximum-leaf spanning tree of the graph in (a).

Since w is in $\text{adj}(R)$ and w is not in P_i, we know that the degree of R is at least two. Therefore $\text{charge}_{T'}(R) \geq 1$, since R cannot be an external normal region.

- $w \in R$

  Since $w \neq v_i$ and $w \neq v_k$, it follows from Lemma 2 that w is not a leaf of T'. We therefore know that $\text{charge}_{T'}(R) \geq 1$.

  If u is in $\text{adj}(R)$, then P_i does not contain any leaf of T', and thus P_i is deficient in T'. If u is not in $\text{adj}(R)$, then uw encloses a node in $\text{adj}(R)$. It follows from P1 and P2 that w is a special leaf of T. Therefore R is special region of T, and thus $\text{charge}_{T'}(R) = 2$. It suffices to show that P_i contains at most one leaf of T'. Assume for a contradiction that P_i contains two leaves of T'. Since we know that every node in $\text{path}_T(u_i, u)$ is not a leaf of T', it follows that both leaves of T' in P_i are enclosed by uw. The 1-optimality of T is contradicted by P1, since u is internal in T.

Now we prove the second statement of the lemma. Assume for a contradiction that $P_j = P_i$ for some $1 \leq j \neq i \leq k-1$. We know that P_i is incident to R_i and P_j is incident to R_j. Since $R_i \neq R_j$, we know that P_i must be incident to R_i and R_j. Therefore R is invalid in T', since $R = R_j$. It has been shown above that w is in $\text{adj}(R) \cup R$. Therefore $w \neq u_i$, $w \neq v_k$. Since w is in $\text{path}_{T'}(u_i, v_k)$, it follows from Lemma 2 that w is internal in T'. We know that R does not contain any normal leaf of T and every special leaf of T in R, if any, is also a leaf in T'. It follows that w must be an internal node in T. Thus u_j, the only node in $P_i \cap \text{adj}(R)$, must be u, since otherwise uw would enclose u_j, and the 1-optimality of T would be contradicted by P3. It follows that u_j is in P_i, and thus u_j is in $\text{path}_{T'}(u_i, v_k)$. If we switch the role of R_i and R_j, know that u_i is in $\text{path}_{T'}(u_j, v_k)$ for exactly the same reason as above. However this contradicts the fact that T' is a spanning tree of G. $\square$

It follows from Claim 2 that we have identified $k-1$ distinct long 2-paths of T that are deficient in T'. This completes the proof of Lemma 8.

5.6 Tightness of the Performance Ratio

We conclude this section by presenting an example of 1-LOT such that the bound of 5 in Theorem 1 is approachable arbitrarily closely. It is easy to verify that the tree in Figure 6-(a) is a 1-LOT. Since the structure is symmetric and repetitive, we only need to verify four different kinds of non-tree edges as candidates for 1-improvements. Since none of these three types of non-tree edges can be involved in any 1-improvement, the tree is a 1-LOT. The maximum-leaf spanning tree for this graph is shown in Figure 6-(b). Clearly the longer this 1-LOT is, the closer to five the ratio is.

6 Performance Guarantee of 2-LOTs

In this section we prove the following restatement of Theorem 2.

Theorem 5 Let T be a 2-LOT of G. Then $|V_1(T')| \leq 3|V_1(T)|$ for any spanning tree T' of G.

If T is a path, Theorem 5 is immediate from Lemma 3. Therefore without loss of generality, we assume that T is not a path for the rest of the section.

Note that we do not use augmentation as we did in the previous section, because the augmentation shown in the previous section might turn a 2-LOT into a tree that is not a 2-LOT, as shown in Figure 4.

We call a short 2-path of T *external* if it is adjacent to a (normal) leaf of T; otherwise we call it *internal*. Let $V_{2E}(T)$ be the set of the external short 2-paths of T. Let $V_{2I}(T)$ be the set of the internal short 2-paths of T. One can easily verify that

$$|V_{2I}(T)| < |\bar{V}_3(T)|, \tag{10}$$

since every internal short 2-path of T is adjacent in T to two nodes in $\bar{V}_3(T)$.

6.1 Basic Strategy

In order to prove Theorem 2, it suffices to show the following lemma.

Lemma 9 Let T be a 2-LOT of G. Then $|V_1(T') \cap (V_{2L}(T) + V_{2E}(T))| \leq |V_1(T) - V_1(T')|$ for any spanning tree T' of G.

Since $V_2(T) = V_{2L}(T) + V_{2E}(T) + V_{2I}(T)$, it follows from Lemma 9 that

$$\begin{aligned} |V_1(T') \cap V_2(T)| &= |V_1(T') \cap (V_{2L}(T) + V_{2E}(T))| + |V_1(T') \cap V_{2I}(T)| \\ &\leq |V_1(T) - V_1(T')| + |V_1(T') \cap V_{2I}(T)| \\ &\leq |V_1(T) - V_1(T')| + |V_{2I}(T)|. \end{aligned} \tag{11}$$

Since $V(G) = V(T) = V_1(T) + V_2(T) + \bar{V}_3(T)$, we have

$$\begin{aligned} |V_1(T')| &= |V_1(T') \cap V_1(T)| + |V_1(T') \cap V_2(T)| + |V_1(T') \cap \bar{V}_3(T)| \\ &\leq |V_1(T') \cap V_1(T)| + |V_1(T') \cap V_2(T)| + |\bar{V}_3(T)| \\ &\leq |V_1(T') \cap V_1(T)| + |V_1(T) - V_1(T')| + |V_{2I}(T)| + |\bar{V}_3(T)| \quad \text{by (11)} \\ &= |V_1(T)| + |V_{2I}(T)| + |\bar{V}_3(T)| \\ &\leq 3|V_1(T)| \quad \text{by (10) and (1).} \end{aligned}$$

Therefore it remains to prove Lemma 9.

6.2 Second Connections, Cross Connections, and Normal Connections

There are three kinds of edges in $G - T$ that could enclose the nodes in $V_{2L}(T) + V_{2E}(T)$.

- An edge uw in $G - T$ is a *second connection of* T if w is in a long 2-path P of T and uw encloses exactly one node of P. It follows from P2 and P3 that u must be a special leaf of T. Moreover P must be incident to the special region of T that contains u. Let $K_S(T)$ denote the set of the second connections of T.

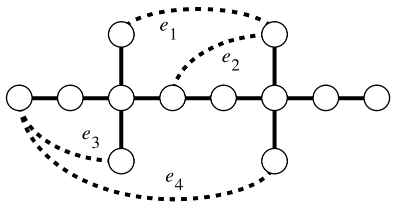


Figure 7: The tree in dark lines is a 2-LOT T. The dashed lines are the non-tree edges, where $e_1 \in K_X(T) - K_N(T)$, $e_2 \in K_S(T)$, $e_3 \in K_N(T) - K_X(T)$, and $e_4 \in K_X(T) \cap K_N(T)$.

- An edge in $G - T$ is a *cross connection of* T if its endpoints are in different regions of T. It follows from P1 and P3 that both endpoints of a cross connection of T must be leaves of T. Let $K_X(T)$ denote the set of the cross connections of T.
- An edge in $G - T$ is a *normal connection of* T if one of its endpoints is in $V_{1N}(T)$. It follows from P2 that the other endpoint of a normal connection of T must be a leaf of T. Note that a normal connection of T could be a cross connection of T. Let $K_N(T)$ denote the set of the normal connections of T.

Let $K(T)$ denote the set of the *connections of* T, which is defined as $K_S(T) \cup K_X(T) \cup K_N(T)$. An example is shown in Figure 7.

We claim that even if we consider only the endpoints of the edges in $K(T) \cap E(T')$, the number of those endpoints that are in $V_1(T) - V_1(T')$ is already no less than that required in Lemma 9. For the rest of the section we prove the following lemma, from which Lemma 9 is immediate.

Lemma 10 Let T be a 2-LOT of G. Suppose T' is a spanning tree of G. Then $|V_1(T') \cap (V_{2L}(T) + V_{2E}(T))| \leq |V(K(T) \cap E(T')) \cap (V_1(T) - V_1(T'))|$.

A node is *green* if it is in $V_1(T') \cap (V_{2L}(T) + V_{2E}(T))$. A node is *yellow* if it is in $V(K(T) \cap E(T')) \cap (V_1(T) - V_1(T'))$. In other words, a green node is a leaf of T' that is in a long 2-path of T or a external short 2-path of T. A yellow node v is an endpoint of an edge of T' that is also a connection of T such that v is a leaf of T' but not a leaf of T. Clearly a green node is not yellow and vice versa, since every green node is a leaf of T', and every yellow node is not a leaf of T'. We prove Lemma 10 by showing that the number of green nodes is no more than the number of yellow nodes.

A connection of T is *yellow* if it has a yellow endpoint. One can easily verify that the (only) yellow endpoint of a yellow second connection of T must be a special leaf of T, since the other endpoint is not a leaf of T.

6.3 Proof of Lemma 10

The proof has the following steps.

Step 1 For every green node v in $V_{2E}(T)$, we show that there exists a yellow normal connection uw of T that encloses v such that (1) u is a normal leaf of T adjacent to v in T and (2) w is yellow. Let y-edge$(v) = uw$ and y-node$(v) = w$. We mark v and w.

Step 2 For every green node v in $V_{2L}(T)$ we do the following. If v is not enclosed by any yellow second connection of T, then we leave v alone. Otherwise let uw be a yellow second connection of T that encloses v, where w is yellow. Let y-edge$(v) = uw$ and y-node$(v) = w$. We mark v and w.

Step 3 For any two distinct unmarked green nodes v and v', we show that y-node$(v) \neq y$-node(v').

Step 4 We show that the number of unmarked green nodes is no more than that of the unmarked yellow nodes.

By the first two steps we know every marked green node has a corresponding marked yellow node. The third step ensures that the number of the marked green nodes is equal to that of the marked yellow nodes. Lemma 10 then follows from the last step. We elaborate these steps in the following subsections.

6.4 Step 1

Let v be a green node in $V_{2E}(T)$. By definition of $V_{2E}(T)$, we know v are adjacent to exactly two nodes, say u and u', in T. Moreover one of u and u', say u, must be a normal leaf of T. Let w be the node such that uw is an edge in $\text{path}_{T'}(u, u')$. Let y-node$(v) = w$. We show w is yellow.

Lemma 11 Let v be a green node in $V_{2E}(T)$. Let y-node(v) be chosen as described above. Then y-node(v) is yellow.

Proof We prove the lemma by showing $w \in V_1(T)$ and $w \notin V_1(T')$.

Assume for a contradiction that uw is an edge of T. It follows that $w = v$, since uv is the only edge of T that is incident to u. Clearly u, v and u' are distinct by the choice of u and u'. It follows from Lemma 2 that w cannot be a leaf of T'. This contradicts the fact that v is green.

Since uw is not an edge of T and u is a normal leaf of T, we know that uw is in $E(T') \cap K_N(T)$. It follows from the property of normal connections that w must be a leaf of T. It suffices to show that w is not a leaf of T'.

Assume for a contradiction that w is a leaf of T'. Since w is in $\text{path}_{T'}(u, u')$, it follows from Lemma 2 that $w = u'$. Since each endpoint of a normal connection of T is a leaf of T, we know u' is a leaf of T. Since v is adjacent in T to only u and u', and u and u' are both leaves of T, it follows that T is equal to the path uvu'. This contradicts the assumption that T is not a path. $\square$

6.5 Step 2

We do not have to prove anything about this step, since it is just an instruction. However we show a lemma about the number of unmarked green nodes in a long 2-path of T when this step is finished.

A long 2-path of T is *green* if it contains unmarked green nodes. A green long 2-path of T is *darkgreen* if it contains more than one green node. It follows from the following lemma that every long 2-path of T contains at most two unmarked green nodes.

Lemma 12 Let v_1 and v_2 be two distinct unmarked green nodes in the same long 2-path of T. Then $v_1 v_2$ is an edge of T.

Proof Let $P = \text{path}_T(v_1, v_2)$. Assume for a contraction that v is a node in P where $v \neq v_1$ and $v \neq v_2$. We show that there exists a yellow second connection of T that encloses one of v_1 and v_2. This contradicts the fact that v_1 and v_2 are both unmarked.

Since T is not a path by assumption, we know $\bar{V}_3(T)$ is not empty. Let v' be a node in $\bar{V}_3(T)$. Let $P' = \text{path}_{T'}(v, v')$. By definition of green nodes, v_1 and v_2 are both leaves of T'. Since v, v',

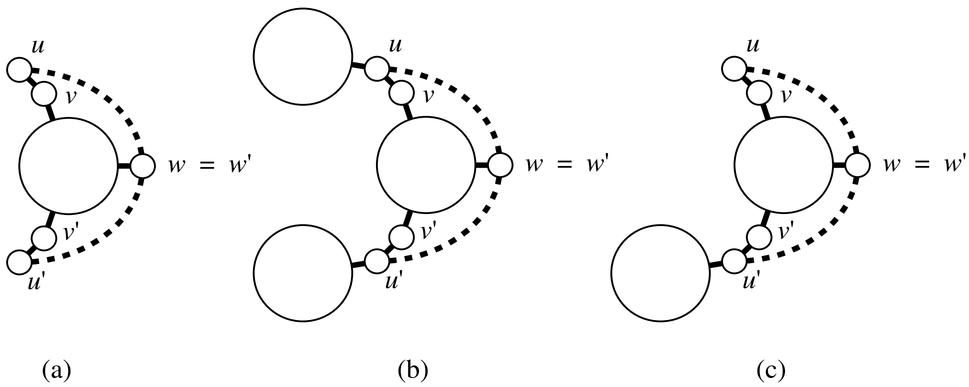


Figure 8:

v_1 and v_2 are all distinct, it follows from Lemma 2 that neither v_1 nor v_2 is in P'. Let w be the node closest to v in P' such that w is not in P. Let u be the node such that uw is an edge in $\text{path}_{T'}(v, w)$. Clearly u is in P and uw is an edge in $T' - T$. Note that v_1, v_2, v, and v' are distinct. Since v_1 and v_2 are leaves of T', it follows from Lemma 2 that P' does not contain v_1 or v_2. By P3 we know that u must be in $\text{path}_T(v_1, v_2)$. Therefore uw encloses either v_1 or v_2. It follows from P1 that uw is a second connection of T. We show that uw is yellow.

Since uw is a second connection of T and $u \in P$, w is a leaf of T by the property of second connections. Since $w \notin \bar{V}_3(T)$, we know $w \neq v'$. Since v, w and v' are distinct and w is in P', it follows from Lemma 2 that w is not a leaf of T'. Therefore w is yellow and so is uw. □

6.6 Step 3

Lemma 13 Let v and v' be two distinct marked green nodes. Then $y\text{-node}(v) \neq y\text{-node}(v')$.

Proof Let $w = y\text{-node}(v)$ and $w' = y\text{-node}(v')$. Let $uw = y\text{-edge}(v)$ and $u'w' = y\text{-edge}(v')$. We prove the lemma for the following four cases. Note that $K_N(T)$ and $K_S(T)$ are disjoint. Therefore $uw \in K_N(T)$ if and only if $v \in V_{2N}(T)$, and $uw \in K_S(T)$ if and only if $v \in V_{2L}(T)$. Similar relations between $u'w'$ and v' also holds.

1. $uw, u'w' \in K_N(T)$
   We know v and v' are both in $V_{2N}(T)$. If $uw = u'w'$, then $u = w'$ and $u' = w$ by the way y-node(v) and y-node(v') are chosen in Step 1. Therefore $w \neq w'$. If $uw \neq u'w'$, assume for a contradiction that $w = w'$. One can easily verify that $(\{uw, u'w'\}, \{uv, u'v'\})$ is a 2-improvement for T, as shown in Figure 8-(a). This contradicts the fact that T is a 2-LOT.

2. $uw, u'w' \in K_S(T)$
   By definition every second connection of T encloses exactly one node in $V_{2L}(T)$. Since $v \neq v'$, we know $uw \neq u'w'$. Assume for a contradiction that $w = w'$. It follows that $(\{uw, u'w'\}, \{uv, u'v'\})$ is a 2-improvement for T, as shown in Figure 8-(b). This contradicts the fact that T is a 2-LOT.

3. $uw \in K_N(T)$ and $u'w' \in K_S(T)$
   Since $K_N(T)$ and $K_S(T)$ are disjoint, we know $uw \neq u'w'$. Assume for a contradiction that $w = w'$. One can easily verify that $(\{uw, u'w'\}, \{uv, u'v'\})$ is a 2-improvement for T, as shown in Figure 8-(c). This contradicts the fact that T is a 2-LOT.

4. $uw \in K_S(T)$ and $u'w' \in K_N(T)$
   The lemma can be proved in a way similar to case 3.

□

6.7 Step 4

In this subsection we prove the following lemma.

Lemma 14 The number of unmarked green nodes of T is at most the number of unmarked yellow nodes of T.

Clearly Lemma 14 holds trivially if T does not have any green long 2-path. Suppose $P_1, \ldots, P_m$ are the green long 2-paths of T for some $m > 0$. We prove Lemma 14 by identifying a yellow cross connection e'_i of T for every $1 \le i \le m$ such that:

E1 If P_i is darkgreen, then both endpoints of e'_i are yellow and unmarked; Otherwise at least one endpoint of e'_i is yellow and unmarked.

E2 If $1 \le i \ne j \le m$, then e'_i and e'_j do not share any common endpoint.

In order to identify those yellow cross connections of T, we are going to find m spanning trees of G, denoted $T_1, T_2, \ldots, T_m$, that satisfy the following property.

T1 $T_1 = T'$.

T2 $T_{i+1} = T_i(\{e_i\}, \{u_i w_i\})$ for every $1 \le i \le m-1$, where (i) e_i is an edge of T such that P_i contains at least one endpoint of e_i, (ii) $u_i w_i$ is a yellows cross connection of T that encloses all nodes in P_i, and (iii) e_i is in $\mathrm{path}_T(u_i w_i)$.

We need the following lemmas to describe the procedure for identifying those m spanning trees and m yellow cross connections of T.

Lemma 15 Suppose P_i is darkgreen for some $1 \le i \le m$. Suppose $T_1, \ldots, T_i$ are spanning trees of G that satisfy T1 and T2. Let v and v' be the unmarked green nodes in P_i. Then $\mathrm{path}_{T_i}(v, v')$ contains a yellow cross connection e'_i of T that encloses v and v'. Moreover both endpoints of e'_i are yellow and unmarked.

Lemma 16 Suppose P_i is green for some $1 \le i \le m$. Suppose $T_1, \ldots, T_i$ are spanning trees of G that satisfy T1 and T2. Let v be an unmarked green node in P_i. Suppose v' and v'' are the nodes adjacent to v in T. Then either $\mathrm{path}_{T_i}(v, v')$ or $\mathrm{path}_{T_i}(v, v'')$ contains a cross connection e'_i of T that encloses v. Moreover at least one endpoint of e'_i is yellow and unmarked.

Here is the procedure for identifying those m spanning trees and m yellow cross connections of T:

Let $T_1 := T'$.
For $i := 1$ to m do
  If P_i is darkgreen then
    Let v and v' be the unmarked green nodes in P_i.
    Let e'_i be a cross connection of T in $\mathrm{path}_{T_i}(v, v')$ that encloses v and v'.

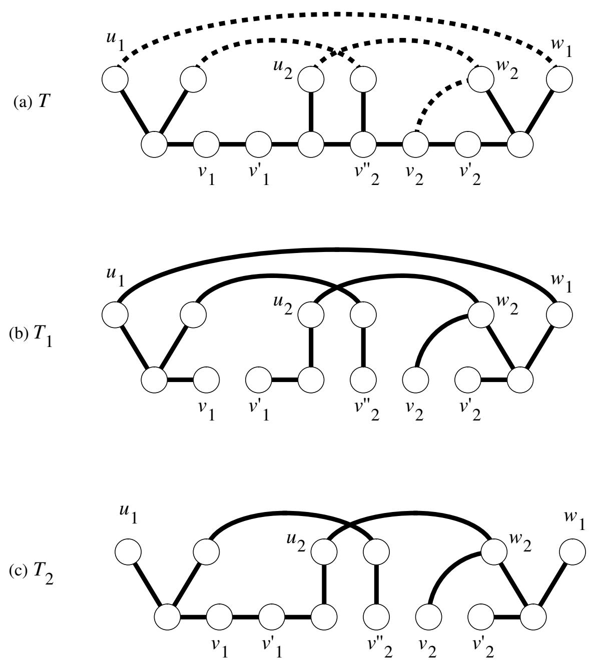


Figure 9:

Let $e_i := vv'$.
else (i.e. P_i is green but not darkgreen)
Let v be the unmarked green node in P_i.
Let v' and v'' be the nodes adjacent to v in T.
Let e'_i be a cross connection of T in $\text{path}_{T_i}(v, v') \cup \text{path}_{T_i}(v, v'')$ that encloses v.
If e'_i is in $\text{path}_{T_i}(v, v')$ then
Let $e_i := vv'$.
else (i.e. e'_i is in $\text{path}_{T_i}(v, v'')$)
Let $e_i := vv''$.
If $i < m$ then
Let $T_{i+1} := T_i(\{e_i\}, \{e'_i\})$.

An example is shown in Figure 9. The dark lines in (a) forms a 2-LOT T, where the dashed lines are the connections of T. Let the picture in (b) be T', which is also known as T_1 in the the procedure. Note that v''_2 is leaf of T', but it is not green, since it is not in a long 2-path of T. Also note that v'_2 is a marked green node of T , where $y\text{-node}(v'_2) = w_2$ and $y\text{-edge}(v'_2) = v_2w_2$. Therefore v_1, v'_1, and v_2 are the unmarked green nodes of T. By definition of green long 2-paths, $v_1v'_1$ is darkgreen and $v_2v'_2$ is green. Since u_1w_1 is the only cross connection of T in $\text{path}_{T'}(v_1, v'_1)$ that encloses v_1 and v'_1, we have that $e'_1 = u_1w_1$ and $e_1 = v_1v'_1$. It follows that T_2 is the tree in (c). Since u_2w_2 is the only cross connection of T in $\text{path}_{T_2}(v'_2, v''_2)$ that encloses v, we have that $e'_2 = u_2w_2$. Since e'_2 is in $\text{path}_{T_2}(v_2, v''_2)$ but not in $\text{path}_{T_2}(v, v'_2)$, it follows that $e_2 = v_2v''_2$. Clearly both endpoints of e'_1 are yellow and unmarked, and P_1 is darkgreen; one endpoint of e'_2 is yellow and unmarked, and P_2 is green but not darkgreen.

Proof of Lemma 14

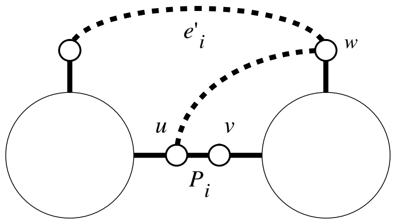


Figure 10:

A. We first prove that the $e'_1, \ldots, e'_m$ described in the procedure do exist and satisfy E1. This can be shown by proving the following claim for every $1 \leq i \leq n$.

- The e'_i described in the i^{th} iteration exists and satisfies E1.
- The $T_1, \ldots, T_{i+1}$ described in the first i iterations are spanning trees of G that satisfy T1 and T2.

We prove the claim by induction on i. When $i = 1$, $T_i = T'$. Clearly T_1 satisfies T1 (and T2). It follows from Lemma 15 and Lemma 16 that the e'_1 described in the first iteration of the procedure does exist. Clearly e_1 satisfies (i) of T2, since v, which is in P_1 in the first iteration, is an endpoint of e_1. By Lemma 15 and Lemma 16 we know that e'_1 satisfies (ii) of T2. One can verify that e'_1 and e_1 satisfy (iii) of T2, since e'_1 is a cross connection of T that encloses all nodes in P_1. It follows from (iii) of T2 that T_2 is a spanning tree of G. Therefore T_1 and T_2 are spanning trees of G that satisfy T1 and T2.

Assume the claim holds for every $1 \leq i < k$ for some $k > 1$. We show that the claim holds for $i = k$. By the inductive hypothesis we know that $T_1, \ldots, T_k$ are spanning trees of G that satisfy T1 and T2. By Lemma 15 and Lemma 16 we know that the e'_k described in the k^{th} iteration of the procedure does exist. Clearly e_k satisfies (i) of T2, since v, which is in P_k in the k^{th} iteration, is an endpoint of e_k. By Lemma 15 and Lemma 16 we know that e'_k satisfies (ii) of T2. One can verify that e'_k and e_k satisfies (iii) of T2, since e'_k is a cross connection of T that encloses all nodes in P_k. It follows from (iii) of T2 that T_{k+1}, which is $T_k(e_k, e'_k)$, is a spanning tree of G. Therefore $T_1, \ldots, T_{k+1}$ are spanning trees of G that satisfy T1 and T2.

B. We show $e'_1, \ldots, e'_m$ satisfy E2. Let i and j be two distinct indices, where $1 \leq i < j \leq m$. We first show that $e'_i \neq e'_j$. By definition of the procedure we know that e'_i is not an edge of T_{i+1}. Since each of $e_{i+1}, \ldots, e_{j-1}$ is an edge of T, and e'_i is not an edge of T, it follows that e'_i is still not an edge of T_j. Note that e'_j is an edge of T_j. Therefore $e'_i \neq e'_j$.

Assume for a contradiction that e'_i and e'_j share a common endpoint. Let e''_i be an edge in P_i and let e''_j be an edge in P_j. One can easily verify that $(\{e'_i, e'_j\}, \{e''_i, e''_j\})$ is a 2-improvement for T, since $P_i \neq P_j$. The 2-optimality of T is contradicted. Therefore E2 is satisfied by $e'_1, \ldots, e'_m$. The lemma is proved. $\square$

The following lemma, which is illustrated in Figure 10, is useful to proving Lemma 15 and Lemma 16.

Lemma 17 Let e'_i be a cross connection of T that encloses the nodes of a green long 2-path P_i of T. Suppose w is a marked yellow endpoint of e'_i. Let u and v be the nodes such that $w = y\text{-node}(v)$ and $uw = y\text{-edge}(v)$. Then uw is a second connection of T that encloses v, and both u and v are in P_i. Moreover u and v are the only two nodes in P_i.

Proof Assume for a contradiction that u is not in P_i. Then v is not in P_i by the way that y-node(v) and y-edge(v) are chosen in Step 1 and Step 2. It follows that $(\{e'_i, uw\}, \{e, uv\})$ is a 2-improvement for T, where e could be any edge in P_i. The 2-optimality of T is thus contradicted. Therefore we know that u is in P_i, which implies that v is also in P_i, and uw is a second connection of T that encloses v.

By definition of second connections we know that uv is an edge of P_i. Let v' be the node other than v that is adjacent to u in T. Let u' be the node other than u that is adjacent to v. In other words, $v'uvu'$ is a path of T. We know that v' is not in P_i, since otherwise $(\{e'_i, uw\}, \{uv, uv'\})$ would be a 2-improvement for T, and the 2-optimality of T would be contradicted. By definition of of second connections we know that u' is not in P_i, since otherwise uw would enclose two nodes, i.e. v and u', of P_i. Since P_i does not contain u' and v', it follows that u and v are the only two nodes in P_i. $\square$

It remains to prove Lemma 15 and Lemma 16.

Proof of Lemma 15 By definition of green nodes, v and v' are both leaves of T'. One can easily verify that P_i does not contain the endpoints of e_j and e'_j, for every $j = 1, \ldots, i-1$. Therefore v and v' are both leaves of T_i.

Let $P' = \text{path}_{T'}(v, v')$. Clearly v and v' are not adjacent in T_i, since otherwise they have to be the only two nodes of G, which contradicts the assumption that T is not a path. By Lemma 12 we know that v and v' are adjacent in T. In order to connect v and v' in T_i, P' must contain either a second connection of T that encloses one of v and v', or a cross connection of T that encloses both v and v'.

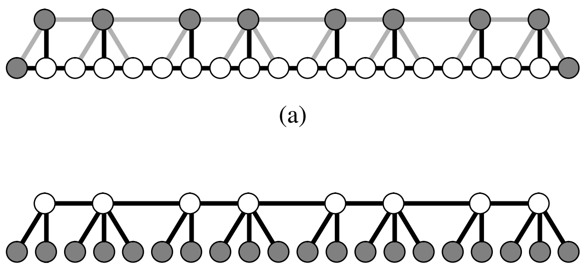
Assume for a contradiction that P' contains a second connection uw of T that encloses one of v and v', say v. By definition of second connections, one endpoint of uw, say u, is a special leaf of T. Since u is in P', u is not a leaf of T_i. One can easily verify that the degree of u in T_i is no more than that in T', since u is not an endpoint of $e_1, \ldots, e_{i-1}$. It follows that u is not a leaf of T'. Therefore uw is a yellow second connection of T that encloses v. This contradicts the fact that v is an unmarked green nodes. Therefore P' has to contain a cross connection e'_i of T that encloses v and v'.

We show that each endpoint of e'_i is not a leaf of T'. Assume for a contradiction that an endpoint u of e'_i is a leaf of T'. If u is not an endpoint of e_1, then u is a leaf of T_2. If u is an endpoint of e_1, we know u must be a normal leaf of T, since e_1 is an edge of T and the other endpoint of e_1 is in P_1, a long 2-path of T. Hence u must be an endpoint of e'_1, and thus u is still a leaf of T_2. For exactly the same reason, we know that u is a leaf of $T_3, \ldots, T_i$. This contradicts the fact that u is in P' by Lemma 2, since $u \neq v$, $u \neq v'$, and $P' = \text{path}_{T'}(v, v')$. Therefore both endpoints of e'_i are yellow.

We show that both endpoints of e'_i are unmarked. Assume for a contradiction that an endpoint w of e'_i is marked. It follows from Lemma 17 that P_i contains only two nodes, and one of them (i.e. the v in Lemma 17) is green and marked. This contradicts the fact that P_i is darkgreen. $\square$

Proof of Lemma 16 Let $P' = \text{path}_{T_i}(v', v'')$. Clearly $E(P') \subseteq E(\text{path}_{T_i}(v, v') \cup \text{path}_{T_i}(v, v''))$. We show that P' contains a cross connection e'_i of T such that e'_i encloses v and at least one endpoint of e'_i is yellow and unmarked.

By definition of green nodes, v is a leaf of T'. One can easily verify that P_i does not contain any endpoints of e_j and e'_j, for every $1 \leq j \leq i-1$. Therefore v is also a leaf of T_i. It follows that



(a)

(b)

Figure 11:

v is not in P' by Lemma 2, since $v \neq v'$ and $v \neq v''$. In order to connect v' and v'' in T_i, P' must contain either a second connection of T that encloses v, or a cross connection of T that encloses v.

Assume for a contradiction that P' contains a second connection uw of T that encloses v. By the property of second connections, one endpoint of uw, say u, is a special leaf of T. Therefore v', u, and v'' are distinct, and thus u is not a leaf of T_i by Lemma 2. Since u is a special leaf of T, we know that u is not an endpoint of $e_1, \ldots, e_{i-1}$. Therefore the degree of u in T_i is no more than that in T'. It follows that u is not a leaf of T'. Therefore uw is a yellow second connection of T that encloses v. This contradicts the fact that v is an unmarked green node. Therefore P' must contain a cross connection e'_i of T that encloses v.

We show that an endpoint of e'_i is not a leaf of T'. Since P_i is a long 2-path of T and v is in P_i, at least one of v' and v'', say v', is in P_i. It follows that v' is not an endpoint of e'_i. Since e'_i is in $\text{path}_{T'}(v', v'')$, it follows from Lemma 2 that at least one endpoint w of e'_i is not a leaf of T_i. We show that w is not a leaf of T' as follows. Let us consider T_{i-1} first. If w is not an endpoint of e_{i-1}, then w is not a leaf of T_{i-1}. If w is an endpoint of e_{i-1}, we know that w must be a normal leaf of T, since e_{i-1} is an edge of T and the other endpoint of e_{i-1} is in P_{i-1}, a long 2-path of T. It follows that w is also an endpoint of e'_{i-1}. Therefore w is not a leaf of T_{i-1}. For exactly the same reason we know that w is not a leaf of $T_{i-1}, T_{i-2}, \ldots, T_1$. Therefore w is yellow.

If w is unmarked, then the lemma is proved. Suppose w is marked. We show that u is yellow and unmarked, where u is the endpoint of e'_i other than w. Since w is yellow and unmarked, it follows from Lemma 17 that P_i contains only two nodes, which must be v and v', where vw is a second connection of T that encloses v'. Therefore v is the only node in P_i that is green and unmarked.

Assume for a contradiction that u is not yellow. Since u is a leaf of T', it follows from Lemma 2 that $u = v''$. Therefore e'_i, which is $v''w$, is a normal connection of T, since u has to be a normal leaf of T. One can easily verify that $(\{e'_i, vw\}, \{vv', vv''\})$ is a 2-improvement for T, which contradicts the 2-optimality of T. Therefore u is yellow.

Assume for a contradiction that u is marked. It follows from Lemma 17 that $y\text{-node}(v) = u$. This contradicts the fact that v is green and unmarked. $\square$

6.8 Tightness of the Performance Ratio

Figure 11 shows the 2-LOT that we believe to have the worst performance ratio, which is only 2.5. Therefore we conjecture the following.

Conjecture 2 The performance guarantee of 2-LOTs is 2.5.

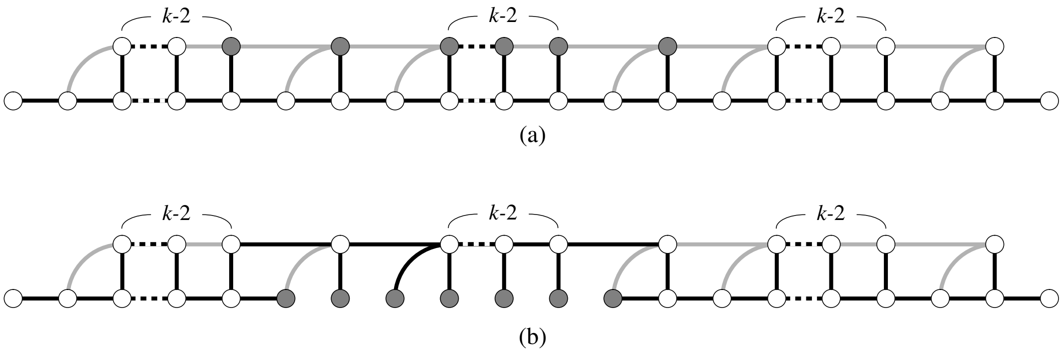


Figure 12: The tree in (a) is an example of k-LOT with performance ratio $\frac{k+1}{k-1}$. This tree is not a $(k+1)$-LOT since the tree in (b) can be obtained by applying a $(k+1)$-improvement.

7 Concluding Remarks

We have discovered a series of examples of graphs in which the performance ratio of a k-LOT can be as bad as $\frac{k+1}{k-1}$ for each $k \geq 3$. We present a typical such graph in Figure 12. However, we have no proof that this is the worst possible performance of k-LOTs on any graph. Since Maximum Leaf Spanning Tree is MAX SNP-complete [8], there must be a constant $\epsilon_0 > 0$ such that there is no $(1+\epsilon_0)$-approximation for Maximum Leaf Spanning Tree, unless P=NP. It follows that the graph shown in Figure 12 is certainly not the worst example of k-LOTs for some $k \geq 3$. It remains open to determine what the ϵ_0 is and construct examples of k-LOTs whose performance ratio is worse than $\frac{k+1}{k-1}$ for some $k \geq 3$.

We mention in passing that the directed version of the problem we considered in which where we seek an arborescence rooted at some fixed node with the maximum number of leaves is also NP-complete. This can be easily deduced using a reduction from the undirected problem. We note that locally optimal algorithms like those we presented for the undirected case perform badly in the directed case.

Consider the closely related Minimum Leaf Spanning Tree problem where we seek a spanning tree with the minimum number of leaves. It is easy to see that this problem is NP-complete by a simple reduction from the Hamiltonian path problem. It is natural to ask whether the techniques similar to ours can be used to approximate this problem as well. We can use recent results on hardness of approximating the longest path problem [13] to infer that the Minimum Leaf Spanning Tree problem is hard to approximate. The key observation is that a spanning tree on n nodes with at most ℓ leaves contains a path with at least $2n/\ell$ edges. This can be shown by considering an Eüler tour of the tree and estimating the longest length of a path between any two consecutive leaves in the tour. A Hamiltonian graph has a spanning tree with at most two leaves. Any polynomial-time approximation algorithm with a multiplicative performance guarantee of f can be used to obtain a spanning tree of a Hamiltonian graph with at most $2f$ leaves. By our observation above, this yields a path in the graph of size at least n/f. The results of Karger, Motwani and Ramkumar [13] imply that f is at least $2^{\Omega(\log^{1-\epsilon} n)}$ for any $\epsilon > 1$ unless $\tilde{\mathrm{P}}$=NP, where $\tilde{\mathrm{P}}$ stands for the complexity class Deterministic Quasi-polynomial time, or DTIME[$n^{\mathrm{polylog}\, n}$].

Acknowledgments

We gratefully acknowledge the support, guidance and encouragement of our advisor, Prof. Philip Klein.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, "Proof verification and the hardness of approximation problems," *Proceedings of the Thirty-third Annual IEEE Symposium on Foundations of Computer Science*, (1992), pp. 14–23.

[2] H. L. Bodlaender, "On linear time minor tests and depth first search," *Proceedings of Workshop on Algorithms and Data Structures*, (1989), pp. 577–590.

[3] G. A. Croes, "A method for solving traveling-salesman problems," *Operations Research* 6, (Nov.-Dec. 1958), pp. 791–812.

[4] E. W. Dijkstra, "Self-stabilizing systems in spite of distributed control," *Communications of ACM* 17, (Nov. 1974), pp. 643–644.

[5] M. R. Fellows and M. A. Langston, "On well-partial-order theory and its applications to combinatorial problems of VLSI design," Technical Report CS-88-188, Dept. of Computer Science, Washington State University, 1988.

[6] M. Fürer and B. Raghavachari, "Approximating the minimum degree spanning tree to within one from the optimal degree," *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, (1992), pp. 317–324.

[7] M. Fürer and B. Raghavachari, "Approximating the minimum-degree Steiner tree to within one of optimal", *Journal of Algorithms* 17, (1994), p. 409–423.

[8] G. Galbiati, F. Maffioli and A. Morzenti, "A short note on the approximability of the maximum leaves spanning tree problem," *Information Processing Letters* 52, (1994), pp. 45–49.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability: A guide to the theory of NP-completeness*, W. H. Freeman, San Francisco (1979).

[10] J. R. Griggs, D. J. Kleitman, and A. Shastri, "Spanning trees with many leaves in cubic graphs," *Journal of Graph Theory* 13, (1989), pp. 669–695.

[11] M. M. Halldórsson "Approximating discrete collections via local improvements," *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, (1995), pp. 160–169.

[12] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis, "How easy is local search?" *Journal of Computer and System Sciences* 37, (1988), pp. 79–100.

[13] D. Karger, R. Motwani, and G. D. S. Ramkumar, "On approximating the longest path in a graph," *Proceedings of Workshop on Algorithms and Data Structures*, (1993), pp. 421–432.

[14] R. M. Karp, "Reducibility among combinatorial problems," in R. E. Miller and J. W. Thatcher (eds.), *Complexity of Computer Computations*. Plenum Press, New York (1972), pp. 85–103.

[15] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *BSTJ* 49, (1970), pp. 291–308.

[16] D. J. Kleitman and D. B. West, "Spanning trees with many leaves," *SIAM Journal on Discrete Mathematics* 4, (Feb. 1991), pp. 99–106.

[17] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the Traveling-Salesman Problem," *Operations Research* 21, (1973), pp. 498–516.

[18] H.-I. Lu and R. Ravi, "The power of local optimization: approximation algorithms for maximum-leaf spanning tree," *Proceedings of the Thirtieth Annual Allerton Conference on Communication, Control and Computing*, (Oct. 1992), pp. 533–542.

[19] C. H. Papadimitriou, A. A. Schäffer and M. Yannakakis, "On the complexity of local search (Extended Abstract)," *Proceedings of the Twenty-second Annual ACM Symposium on the Theory of Computing*, (1990), pp. 438–445.

[20] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Inc. (1982).

[21] C. H. Papadimitriou and M. Yannakakis, "Optimization, Approximation, and Complexity classes," *Proceedings of the Twentieth Annual ACM Symposium on the Theory of Computing*, (1988) pp. 229–234.

[22] C. Payan, M. Tchuente, and N. H. Xuong, "Arbres avec un nombres maximum de sommets pendants," *Discrete Mathematics* 49, (1984), pp. 267–273.

[23] A. A. Schäffer and M. Yannakakis, "Simple local search problems that are hard to solve," *SIAM Journal on Computing* 20(1), (1991), pp. 56–87.

[24] R. Ravi, B. Raghavachari, and P. N. Klein, "Approximation through local optimality: Designing networks with small degree," *Proceedings, Twelfth Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, (Dec. 1992), LNCS 652, pp. 279–290.

[25] J. A. Storer, "Constructing full spanning trees for cubic graphs," *Information Processing Letters* 13, (1981), pp. 8–11.

[26] M. Tchuente, "Sur l'auto-stabilisation dans un réseau d'odinateurs," *R. A. I. R. O. Informatique Théorique* 15, (1981), pp. 47–66.

[27] M. Yannakakis, "The analysis of local search problems and their heuristics," *STACS*, (1990), pp. 289–311 .