

# Approximation algorithms for quickest spanning tree problems

Refael Hassin<sup>1</sup> and Asaf Levin<sup>2</sup>

June 20, 2004

## Abstract

Let  $G = (V, E)$  be an undirected multi-graph with a special vertex  $root \in V$ , and where each edge  $e \in E$  is endowed with a length  $l(e) \geq 0$  and a capacity  $c(e) > 0$ . For a path  $P$  that connects  $u$  and  $v$ , the *transmission time* of  $P$  is defined as  $t(P) = \sum_{e \in P} l(e) + \max_{e \in P} \frac{1}{c(e)}$ . For a spanning tree  $T$ , let  $P_{u,v}^T$  be the unique  $u - v$  path in  $T$ . The QUICKEST RADIUS SPANNING TREE PROBLEM is to find a spanning tree  $T$  of  $G$  such that  $\max_{v \in V} t(P_{root,v}^T)$  is minimized. In this paper we present a 2-approximation algorithm for this problem, and show that unless  $P = NP$ , there is no approximation algorithm with performance guarantee of  $2 - \epsilon$  for any  $\epsilon > 0$ . The QUICKEST DIAMETER SPANNING TREE PROBLEM is to find a spanning tree  $T$  of  $G$  such that  $\max_{u,v \in V} t(P_{u,v}^T)$  is minimized. We present a  $\frac{3}{2}$ -approximation to this problem, and prove that unless  $P = NP$  there is no approximation algorithm with performance guarantee of  $\frac{3}{2} - \epsilon$  for any  $\epsilon > 0$ .

## 1 Introduction

Let  $G = (V, E)$  be an undirected multi-graph where each edge  $e \in E$  is endowed with a length  $l(e) \geq 0$  and a capacity  $c(e) > 0$ . For an edge  $e = (u, v) \in E$ , the *capacity* of  $e$  is the amount of data that can be sent from  $u$  to  $v$  along  $e$  in a single time unit. The *reciprocal capacity* along  $e$  is  $r(e) = \frac{1}{c(e)}$ . The *length* of  $e$  is the time required to send data from  $u$  to  $v$  along  $e$ . If we want to transmit a data of size  $\sigma$  along  $e$ , then the time it takes is the *transmission time*,  $t(e) = l(e) + \frac{\sigma}{c(e)}$ . For a path  $P$  that connects  $u$  and  $v$  the *length* of  $P$  is defined as  $l(P) = \sum_{e \in P} l(e)$ , the *reciprocal capacity* of  $P$  is defined as  $r(P) = \max_{e \in P} r(e)$ , and the *transmission time* of  $P$  is defined as  $t(P) = l(P) + \sigma r(P)$ . Denote  $n = |V|$  and  $m = |E|$ .

A path  $P$  that connects  $u$  and  $v$  is a  $u - v$  path. For a spanning tree  $T$ , let  $P_{u,v}^T$  be the unique  $u - v$  path in  $T$ .

The QUICKEST PATH PROBLEM (QPP) is to find for a given pair of vertices  $u, v \in V$  a  $u - v$  path  $P$ , whose transmission time is minimized. Moore [8] introduced this problem and presented an  $O(m^2 + mn \log n)$ -time algorithm that solves the problem for all possible values of  $\sigma$ . Chen and Chin [4] showed an  $O(m^2 + mn \log n)$ -time algorithm that solves the problem for a common

---

<sup>1</sup>Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel. hassin@post.tau.ac.il

<sup>2</sup>Faculty of Industrial Engineering and Management, The Technion, Haifa 32000, Israel. levinas@tx.technion.ac.il

source  $u$  and all destinations  $v$  for all possible values of  $\sigma$ . Their algorithm creates a data structure that encodes the quickest paths for all possible values of  $\sigma$ . Using this data-structure, it is possible to obtain an optimal path for a single value of  $\sigma$  in  $O(\log n)$  time (after the  $O(m^2 + mn \log n)$ -time preprocessing). Martins and Santos [7] provided a different algorithm for the same problem with the same time-complexity and a reduced space complexity. Rao [10] considered the problem with a single value of  $\sigma$  and a given pair  $u, v$ . He designed a probabilistic algorithm with running time  $O(pm + pn \log n)$  that has a high probability (that depends on  $p$ ) of computing a path with transmission time close to the optimal one. The all-pairs shortest paths variant of this problem is solved in  $O(mn^2)$  time (after the  $O(m^2 + mn \log n)$ -time preprocessing) for all possible values of  $\sigma$  and moreover, the algorithm constructs a data structure such that it is possible to get an optimal path for a single value of  $u, v$  and  $\sigma$  in  $O(\log n)$  time. This algorithm to solve the all-pairs variant appeared in both Chen and Hung [1] and Lee and Papadopoulou [6]. Rosen, Sun and Xue [11] and Pelegrin and Fernandez [9] considered the bi-criteria minimization problem of finding a  $u - v$  path whose length and reciprocal capacity are both minimized. These objectives are sometimes contradicting, and [11, 9] presented algorithms that compute the efficient path set. Chen [2, 3] and Rosen, Sun and Xue [11] considered the  $k$  quickest simple path problem.

In this paper we consider problems with a single value of  $\sigma$ . W.l.o.g. we can assume that  $\sigma = 1$  (otherwise we can scale  $c$ ). Therefore, for a  $u - v$  path  $P$ ,  $t(P) = l(P) + r(P)$ .

Many network design problems restrict the solution network to be a tree (i.e., a subgraph without cycles). Usually such a constraint results from the communication protocols used in the network. In broadcast networks, one usually asks for a small maximum delay of a message sent by a *root* vertex to all the users (all the other vertices) of the network. In other communication networks, one seeks a small upper bound on the delay of transmitting a message between any pair of users (vertices). These objectives lead to the following two problems:

The QUICKEST RADIUS SPANNING TREE PROBLEM is to find a spanning tree  $T$  of  $G$  such that  $rad_t(T) = \max_{v \in V} t(P_{root,v}^T)$  is minimized.

The QUICKEST DIAMETER SPANNING TREE PROBLEM is to find a spanning tree  $T$  of  $G$  such that  $diam_t(T) = \max_{u,v \in V} t(P_{u,v}^T)$  is minimized.

The *shortest paths tree* (SPT) is defined as follows: given an undirected multi-graph  $G = (V, E)$  with a special vertex  $root \in V$ , and where each edge  $e \in E$  is endowed with a length  $l(e) \geq 0$ , the SPT is a spanning tree  $T = (V, E_T)$  such that for every  $u \in V$  the unique  $root - u$  path in  $T$

has the same total length as the length of the shortest path in  $G$  that connects  $root$  and  $u$ . This tree can be found in polynomial time by applying Dijkstra's algorithm from  $root$ . We denote the returned tree by  $SPT(G, root, l)$ .

To define the absolute 1-center of a graph  $G = (V, E)$  where each edge  $e \in E$  is endowed with a length  $l(e) \geq 0$ , we refer also to interior points on an edge by their distances (along the edge) from the two end-vertices of the edge. We let  $A(G)$  denote the continuum set of points on the edges of  $G$ .  $l$  induces a length function over  $A(G)$ . For  $x, y \in A(G)$ , let  $l_G(x, y)$  denote the length of a shortest path in  $A(G)$  connecting  $x$  and  $y$ . The *absolute 1-center* is a minimizer  $x$  of the function  $F(x) = \max_{v \in V} l_G(x, v)$ .

The MINIMUM DIAMETER SPANNING TREE PROBLEM is defined as follows: given an undirected multi-graph  $G = (V, E)$  where each edge is endowed with a length  $l(e) \geq 0$ , we want to find a spanning tree  $T = (V, E_T)$  such that  $diam_l(T) = \max_{u, v \in V} l(P_{u, v}^T)$  is minimized. This problem is solved [5] in polynomial time by computing a shortest path tree from an absolute 1-center of  $G$ .

**Our results:**

1. A 2-approximation algorithm for the QUICKEST RADIUS SPANNING TREE PROBLEM.
2. For any  $\epsilon > 0$ , unless  $P = NP$  there is no approximation algorithm with performance guarantee of  $2 - \epsilon$  for the QUICKEST RADIUS SPANNING TREE PROBLEM.
3. A  $\frac{3}{2}$ -approximation algorithm for the QUICKEST DIAMETER SPANNING TREE PROBLEM.
4. For any  $\epsilon > 0$ , unless  $P = NP$  there is no approximation algorithm with performance guarantee of  $\frac{3}{2} - \epsilon$  for the QUICKEST DIAMETER SPANNING TREE PROBLEM.

## 2 A 2-approximation algorithm for the quickest radius spanning tree problem

Algorithm *quick\_radius*:

1. For every  $u \in V \setminus \{root\}$ , compute  $QP_{root, u}$ , that is a quickest path from  $root$  to  $u$  in  $G$ .
2. Let  $G' = (V, E') = \bigcup_{u \in V \setminus \{root\}} QP_{root, u}$ .
3.  $T' = SPT(G', root, l)$ .
4. **Return**  $T'$ .

**Example 1** Consider the graph  $G = (\{root, 1, 2, 3\}, E_1 \cup E_2)$ , where  $E_1 = \{(root, 1), (1, 2)\}$  and for each  $e \in E_1$   $l(e) = 1$ ,  $r(e) = 2$ , and  $E_2 = \{(root, 2), (2, 3)\}$  and for each  $e \in E_2$   $l(e) = 0$  and  $r(e) = 3\frac{1}{2}$ . Then,  $QP_{root,1} = (root, 1)$ ,  $QP_{root,2} = (root, 1, 2)$  and  $QP_{root,3} = (root, 2, 3)$ . Therefore,  $G'$  that is computed in Step 2 of Algorithm *quick\_radius* is exactly  $G$ . In Step 3 we compute the tree  $T' = (\{root, 1, 2, 3\}, E_{T'})$  where  $E_{T'} = \{(root, 1), (root, 2), (2, 3)\}$ , and therefore the cost of  $T'$  is defined by the path  $P = (root, 2, 3)$  where  $r(P) = r(root, 2) = 3\frac{1}{2}$ ,  $l(P) = 1 + 0 = 1$ ,  $t(P) = r(P) + l(P) = 3\frac{1}{2} + 1 = 4\frac{1}{2}$ . Note that  $T'$  is the optimal solution.

The time complexity of Algorithm *quick\_radius* is  $O(m^2 + mn \log m)$ . It is dominated by the time complexity of Step 1 that is  $O(m^2 + mn \log m)$  by Chen and Chin [4]. The algorithm returns a spanning tree of  $G$ , and therefore it is feasible.

Before we prove the performance guarantee of the algorithm we remark that as in Example 1  $G'$  may contain cycles. The reason for this is that the property of shortest paths, “any sub-path of a shortest path must itself be a shortest path” does not hold for quickest paths (as noted in [4]).

**Theorem 2** *Algorithm quick\_radius is a 2-approximation algorithm for the QUICKEST RADIUS SPANNING TREE PROBLEM.*

**Proof:** Denote by  $OPT$  the cost of an optimal solution. Note that for every  $u \in V$ ,  $t(QP_{root,u}) \leq OPT$ . In particular, for every  $e \in E'$ ,  $r(e) \leq OPT$ , and for every  $u \in V$ ,  $l(QP_{root,u}) \leq OPT$ . Therefore, for every  $u \in V$ ,  $l(P_{root,u}^{T'}) \leq l(QP_{root,u}) \leq OPT$  and  $\max_{e \in P_{root,u}^{T'}} r(e) \leq \max_{e \in E'} r(e) \leq OPT$ . Therefore,

$$t(P_{root,u}^{T'}) = l(P_{root,u}^{T'}) + \max_{e \in P_{root,u}^{T'}} r(e) \leq 2OPT.$$

Since the last inequality holds for every  $u \in V$ , it holds also for their maximum,  $rad_t(T')$ . ■

### 3 Inapproximability of the quickest radius spanning tree problem

In this section we show that unless  $P = NP$ , there is no approximation algorithm with a performance guarantee of  $2 - \epsilon$  for any  $\epsilon > 0$ . In order to motivate our final reduction, we first present a simpler reduction that shows that there is no approximation algorithm with a performance guarantee of  $\frac{3}{2} - \epsilon$ . Next, we show an example where the fact that Algorithm *quick\_radius* chooses its output from the edge set of the quickest paths, does not allow it to have a performance guarantee

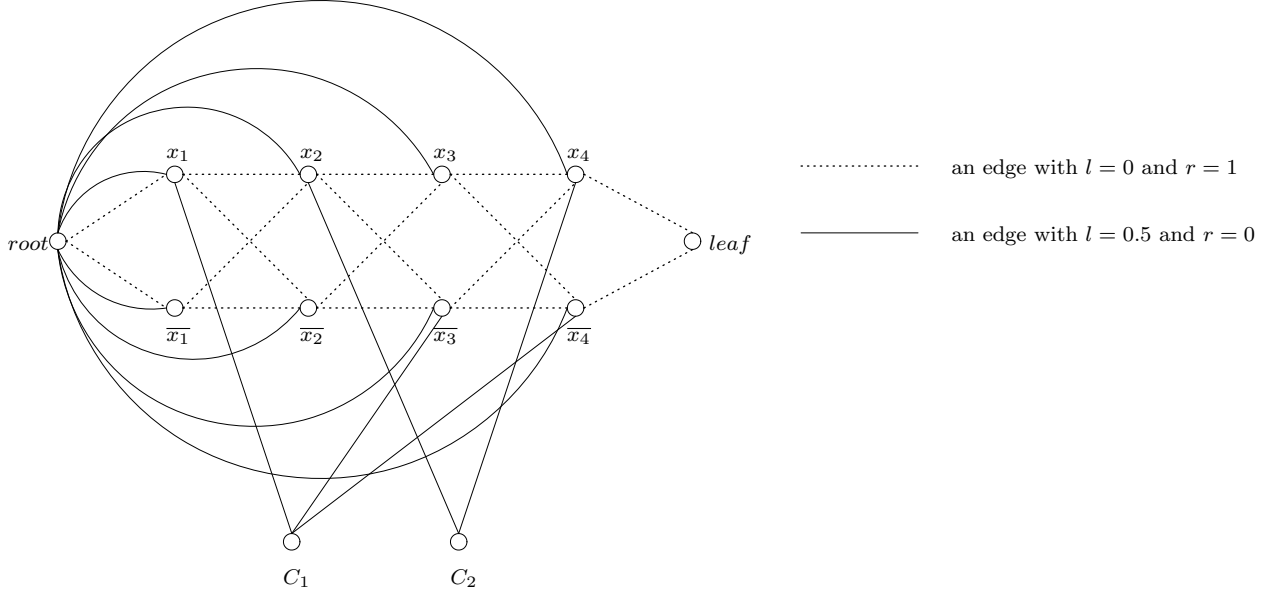


Figure 1: Example for the construction of Lemma 3:  $C_1 = x_1 \vee \bar{x}_3 \vee \bar{x}_4$  and  $C_2 = x_2 \vee x_4$

of better than 2. Finally, we present the main result of this section by combining the ideas of its first two parts.

### 3.1 A $\frac{3}{2}$ lower bound on the approximation ratio

**Lemma 3** *If  $P \neq NP$ , then there is no  $(\frac{3}{2} - \epsilon)$ -approximation algorithm for the QUICKEST RADIUS SPANNING TREE PROBLEM for any  $\epsilon > 0$ .*

**Proof:** We prove the claim via reduction from SAT. Let  $C_1, C_2, \dots, C_m$  be the clauses of a SAT instance in the variables  $x_1, x_2, \dots, x_n$ . We construct a graph  $G = (V, E)$  as follows (see Figure 1):  $V = \{root, leaf\} \cup \{x_i, \bar{x}_i : 1 \leq i \leq n\} \cup \{C_j : C_j \text{ is a clause}\}$ , i.e., we have special pair of vertices  $root$  and  $leaf$ , we have a vertex for each variable and another for its negation, and a vertex for each clause.  $E = E_1 \cup E_2 \cup E_3$ .  $E_1 = \{(root, x_1), (root, \bar{x}_1)\} \cup \{(x_i, x_{i+1}), (x_i, \bar{x}_{i+1}), (\bar{x}_i, x_{i+1}), (\bar{x}_i, \bar{x}_{i+1}) : 1 \leq i \leq n-1\} \cup \{(x_n, leaf), (\bar{x}_n, leaf)\}$ , where for every  $e \in E_1$ ,  $l(e) = 0$  and  $r(e) = 1$ .  $E_2 = \{(l_i, C_j) : \text{for every literal } l_i \text{ of } C_j \forall j\}$ ,  $E_3 = \{(root, x_i), (root, \bar{x}_i) : 1 \leq i \leq n\}$ , where for every  $e \in E_2 \cup E_3$ ,  $l(e) = 0.5$  and  $r(e) = 0$ . This instance contains parallel edges, however it is easy to make it a graph by splitting edges. This defines the instance for the QUICKEST RADIUS SPANNING TREE PROBLEM.

Assume that the formula is satisfied by some truth assignment. We now show how to construct a spanning tree  $T = (V, E_T)$  such that  $rad_t(T) = 1$ .  $E_T = E_T^1 \cup E_T^2$ .  $E_T^1$  is a path of edges from  $E_1$  that connects  $root$  and  $leaf$  where all its intermediate vertices are false literals in the truth assignment.  $E_T^2$  is composed of the edges  $(root, l_i)$  from  $E_3$  for all the true literals  $l_i$ , also each clause contains at least one true literal, and we add to  $E_T^2$  an edge between the clause vertex and such a true literal (exactly one edge for each clause vertex). Then,  $rad_t(T) = 1$  because for every  $u \in V$  that is on the path of  $E_T^1$  edges, we have a path whose transmission time is exactly 1, and for every other vertex we have a path from  $root$  of length at most 1 and reciprocal capacity 0, so that its transmission time is at most 1.

We now show that if the formula cannot be satisfied, then for every tree  $T = (V, E_T)$ ,  $rad_t(T) \geq \frac{3}{2}$ . If  $P_{root,leaf}^T$  contains an edge from  $E_3$  (at least one), then its length is at least  $\frac{1}{2}$ , its reciprocal capacity is 1, and therefore, its transmission time is at least  $\frac{3}{2}$ . Assume now that  $P_{root,leaf}^T$  is composed only of edges from  $E_1$ . Since the formula cannot be satisfied, then for every such path  $P_{root,leaf}^T$  there is a clause vertex  $C_j$  all of whose literals are in  $P_{root,leaf}^T$ . For this  $C_j$ ,  $P_{root,C_j}^T$  has length at least  $\frac{1}{2}$ , and its reciprocal capacity is 1. Therefore, its transmission time is at least  $\frac{3}{2}$ .

For any  $\epsilon > 0$ , a  $(\frac{3}{2} - \epsilon)$ -approximation algorithm distinguishes between 1 and  $\frac{3}{2}$ , and therefore solves the SAT instance. ■

### 3.2 On using only quickest path edges

Recall that  $G'$  is the union of a quickest  $root - u$  path in  $G$  for all  $u \in V \setminus \{root\}$ . Let  $\epsilon > 0$  and denote by  $OPT$  the cost of an optimal solution. In this subsection we show an instance where any spanning tree  $T'$  of  $G'$  satisfies  $rad_t(T') \geq (2 - \epsilon)OPT$ .

Let  $\delta > 0$  be small enough, and let  $k$  be a large enough integer. We construct a graph  $G = (V, E)$  as follows:  $V = \{root, leaf\} \cup \{i, i' : 1 \leq i \leq k - 1\}$ .  $E = E_1 \cup E_2 \cup E_3 \cup E_4$ .  $E_1 = \{(root, 1)\} \cup \{(i, i + 1) : 1 \leq i \leq k - 2\} \cup \{(k - 1, leaf)\}$  where for each edge  $e \in E_1$ ,  $l(e) = 0$  and  $r(e) = 1$ .  $E_2 = \{(root, 1)\} \cup \{(i, i + 1) : 1 \leq i \leq k - 2\}$  where for each edge  $e \in E_2$ ,  $l(e) = \frac{1}{k}$  and  $r(e) = 0$ .  $E_3 = \{(i, i') : 1 \leq i \leq k - 1\}$ , where for each  $e_i = (i, i')$ ,  $l(e_i) = \frac{k-i}{k}$  and  $r(e_i) = 0$ .  $E_4 = \{(root, i') : 1 \leq i \leq k - 1\}$  where for each edge  $e \in E_4$ ,  $l(e) = 1 + \delta$  and  $r(e) = 0$ .

The quickest paths from  $root$  use only the edges from  $E_1 \cup E_2 \cup E_3$ , and the subgraph  $G'$  for  $k = 5$  is shown in Figure 2.

We note that in this graph the tree  $T = (V, E_T)$  where  $E_T = E_1 \cup E_4$ , has  $rad_t(T) = 1 + \delta$ .

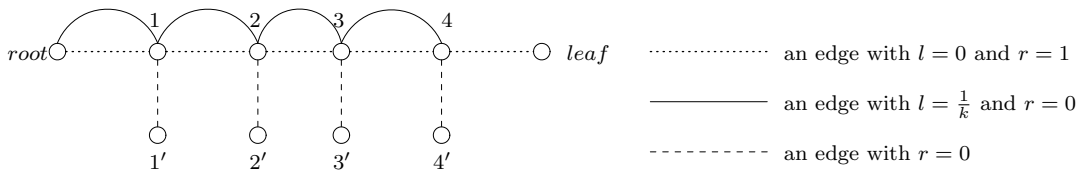


Figure 2: The graph  $G'$  for the example in Subsection 3.2

Consider a spanning tree  $T' = (V, E_{T'})$  where  $E_{T'} \subseteq E_1 \cup E_2 \cup E_3$ . Then, if  $P_{root,leaf}^{T'}$  is composed of all the edges of  $E_2$  and the edge  $(k-1, leaf) \in E_1$ , then  $rad_t(T') \geq 2 - \frac{1}{k}$ . Otherwise,  $P_{root,leaf}^{T'}$  contains other edges from  $E_1$ . Denote  $0 = root$ , and let  $(i-1, i) \in P_{root,leaf}^{T'}$  be an edge from  $E_1$  such that  $i$  is minimized ( $i \geq 1$ ). Then,  $P_{root,i'}^{T'}$  is composed of the edge  $(i, i')$  and the part of  $P_{root,leaf}^{T'}$  that connects  $root$  and  $i$ . Therefore, its length is at least  $1 - \frac{1}{k}$  (by the minimality of  $i$ ), and its reciprocal capacity is 1. Therefore,  $rad_t(T') \geq 2 - \frac{1}{k}$ .

We pick  $\delta$  and  $k$  such that  $\frac{2 - \frac{1}{k}}{1 + \delta} \geq 2 - \epsilon$ .

This example also shows that our analysis of Algorithm *quick\_radius* is tight.

### 3.3 A tight lower bound on the approximability ratio

In this subsection we show that unless  $P = NP$  there is no polynomial time approximation algorithm that guarantees a ratio of  $2 - \epsilon$  for any  $\epsilon > 0$ . Since Algorithm *quick\_radius* is a 2-approximation algorithm, it is the **best possible**.

**Theorem 4** *If  $P \neq NP$ , then there is no  $(2 - \epsilon)$ -approximation algorithm for the QUICKEST RADIUS SPANNING TREE PROBLEM for any  $\epsilon > 0$ .*

**Proof:** We prove the claim via reduction from SAT. Let  $C_1, C_2, \dots, C_m$  be the clauses of a SAT instance in the variables  $x_1, x_2, \dots, x_n$ . Let  $k$  be a large integer (to be chosen later). We construct an instance of the QUICKEST RADIUS SPANNING TREE PROBLEM in the following way (see Figure 3):  $G = (V, E)$  where  $V = \bigcup_{j=1}^{k-1} V_j$ .  $V$  is composed of a special vertex  $root$  and of  $k-1$  levels.  $V_j = \{head^j, tail^j\} \cup \{x_i^j, \overline{x_i^j} : 1 \leq i \leq n\} \cup \{C_i^j : 1 \leq i \leq m\}$ . I.e., each level has special vertices  $head^j$  and  $tail^j$ , and it has a vertex for each variable and another for its negation, and for each clause.  $E = E_1 \cup E_2 \cup E_3$ .

$$E_1 = \{(head^j, x_1^j), (head^j, \overline{x_1^j}) : 1 \leq j \leq k-1\}$$

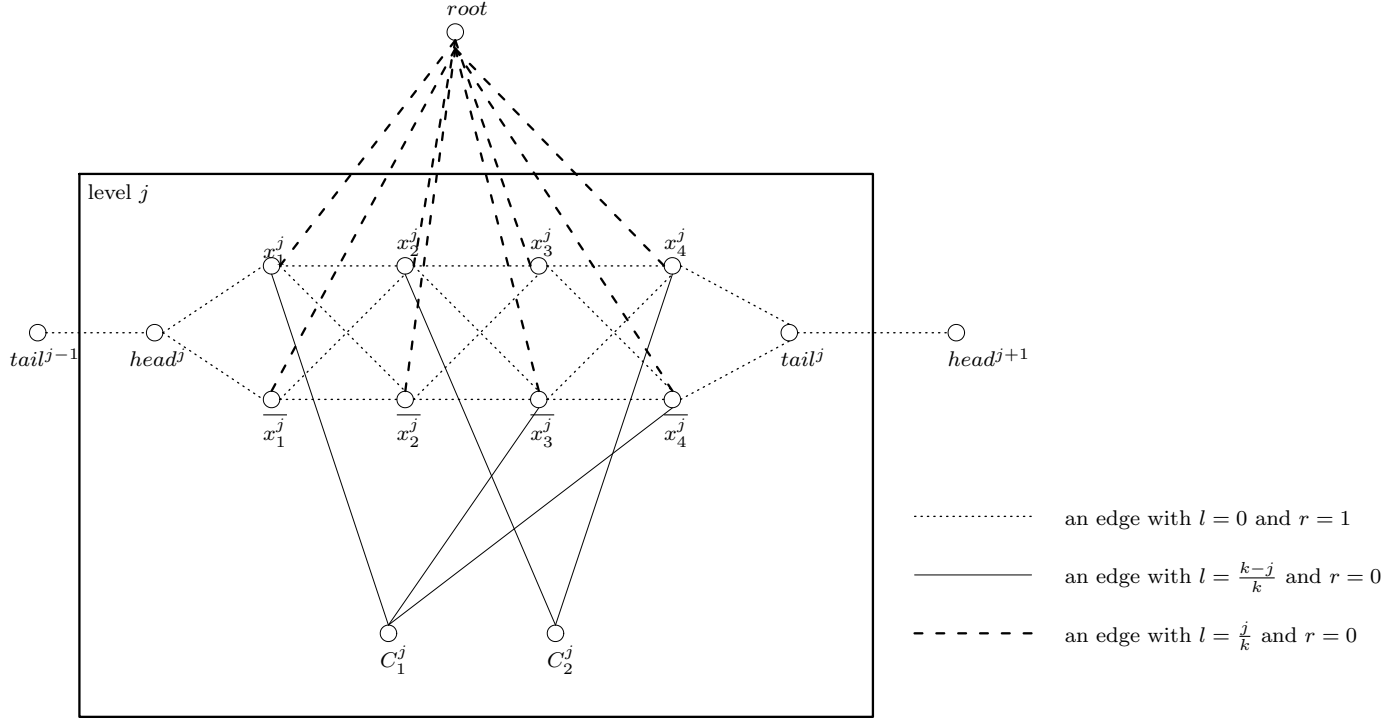


Figure 3: Level  $j$  in the proof of Theorem 4

$$\begin{aligned}
& \cup \{(x_i^j, x_{i+1}^j), (x_i^j, \overline{x_{i+1}^j}), (\overline{x_i^j}, x_{i+1}^j), (\overline{x_i^j}, \overline{x_{i+1}^j}) : 1 \leq i \leq n-1, 1 \leq j \leq k-1\} \\
& \cup \{(x_n^j, \overline{tail^j}), (\overline{x_n^j}, tail^j) : 1 \leq j \leq k-1\} \\
& \cup \{(tail^j, head^{j+1}) : 1 \leq j \leq k-2\}.
\end{aligned}$$

For each edge  $e \in E_1$ ,  $l(e) = 0$  and  $r(e) = 1$ .  $E_2 = \{(literal_i^j, C_p^j) : literal_i \in C_p \forall p\}$ . For an edge  $e = (literal_i^j, C_p^j) \in E_2$ ,  $l(e) = \frac{k-j}{k} = 1 - \frac{j}{k}$  and  $r(e) = 0$ .  $E_3 = \{(root, x_i^j), (root, \overline{x_i^j}) : 1 \leq i \leq n, 1 \leq j \leq k-1\}$ . For an edge  $e = (root, literal_i^j)$  where  $literal_i^j$  is either  $x_i^j$  or  $\overline{x_i^j}$ ,  $l(e) = \frac{j}{k}$  and  $r(e) = 0$ . Note that the length of edges of  $E_2 \cup E_3$  depends on the level of the vertices. The final step in the construction of  $G$  is to identify  $root = head^1$ .

Note that for  $k = 2$  the resulting instance is exactly the one in the proof of Lemma 3.

We first argue that if the SAT formula is satisfied by a truth assignment  $TRUTH$ , then there is a spanning tree  $T = (V, E_T)$  such that  $rad_t(T) = 1$ . Define  $E_T$  as follows: we take a path from  $root$  to  $tail^{k-1}$  of edges from  $E_1$  that traverses through all the literal vertices (i.e.,  $x_i^j$  and  $\overline{x_i^j}$  for all  $i$  and  $j$ ) that are false literals according to  $TRUTH$  (note that this path traverses through the vertices

$head^j$  and  $tail^j$  for all  $j$ ). For every clause  $C_p$ , there is a literal  $literal_i \in C_p$  that is assigned the value  $TRUE$  by  $TRUTH$ , we pick such literal and we add to  $E_T$  the edges  $(literal_i^j, C_p^j) \in E_2$  for all  $j$ . From  $E_3$  we take all the edges  $(root, literal_i^j)$  such that  $TRUTH(literal_i) = TRUE$ . For a vertex  $u$  along the path of edges from  $E_1$ ,  $l(P_{root,u}^T) = 0$  and  $r(P_{root,u}^T) = 1$ . For another literal vertex  $u$  (that is not on the path of edges from  $E_1$ ),  $l(P_{root,u}^T) \leq 1$  and  $r(P_{root,u}^T) = 0$ . For a clause vertex  $u = C_p^j$ ,  $l(P_{root,u}^T) = 1$  and  $r(P_{root,u}^T) = 0$ . Therefore,  $rad_t(T) = 1$ .

It remains to show that if the SAT formula cannot be satisfied, then for every tree  $T = (V, E_T)$ ,  $rad_t(T) \geq 2 - \frac{1}{k}$ . To prove this claim, we first prove an auxiliary claim. Suppose that for every  $q = 1, 2, \dots, j$ ,  $(V, E_T \cap E_1)$  does not contain a path from  $head^q$  to  $tail^q$ , then  $l(P_{root,tail^j}^T) \geq \frac{j}{k}$ . We prove the auxiliary claim by induction over  $j$ . For  $j = 1$ , the claim is trivial since  $P_{root,tail^1}^T$  must have an edge from  $E_2 \cup E_3$  (because by assumption it is not contained in  $E_1$ ). We assume that the claim holds for all the previous levels, and we prove it for  $j$ . By assumption,  $(V, E_T \cap E_1)$  does not contain a  $tail^j - head^j$  path. If  $P_{root,tail^j}^T$  has an edge from  $E_3$  that connects a vertex from a level at least  $j$  to  $root$ , then the claim holds. Otherwise,  $P_{root,tail^j}^T$  must traverse through  $(tail^{j-1}, head^j)$ . Therefore, it consists of two (edge-)disjoint sub-paths: the first connects  $root$  to  $tail^{j-1}$ , and the second connects  $tail^{j-1}$  and  $tail^j$ . By the induction assumption  $l(P_{root,tail^{j-1}}^T) \geq \frac{j-1}{k}$  and by assumption  $l(P_{tail^{j-1},tail^j}^T) \geq \frac{2}{k}$  because it must contain two edges from  $E_2$  and the length of such an edge is at least  $\frac{1}{k}$ . Therefore, the auxiliary claim is proved.

We now use a similar method to the method used in Subsection 3.2 to prove that  $rad_t(T) \geq 2 - \frac{1}{k}$ .

If  $(V, E_T \cap E_1)$  does not contain a path from  $head^j$  to  $tail^j$  for every  $j = 1, 2, \dots, k-1$ , then by the auxiliary claim  $l(P_{root,tail^{k-1}}^T) \geq \frac{k-1}{k}$ , and  $r(P_{root,tail^{k-1}}^T) = 1$ . Therefore,  $rad_t(T) \geq 2 - \frac{1}{k}$ .

Assume otherwise, and let  $j$  be the minimum level such that  $(V, E_T \cap E_1)$  contains a path  $P$  from  $head^j$  to  $tail^j$ . We first assume that  $j \geq 2$ . By the auxiliary claim for  $j-1$  (using the minimality of  $j$ ),  $l(P_{root,tail^{j-1}}^T) \geq \frac{j-1}{k}$ . If  $(tail^{j-1}, head^j) \in E_T$ , then  $l(P_{root,tail^j}^T) \geq \frac{j-1}{k}$ . Otherwise,  $P_{root,tail^j}^T$  has an edge from  $E_3$  that connects a vertex from a level at least  $j$  to  $root$ , and in this case also  $l(P_{root,tail^j}^T) \geq \frac{j-1}{k}$ . If  $j = 1$ , then  $l(P_{root,tail^j}^T) \geq \frac{j-1}{k}$ . Consider the truth assignment of assigning a literal  $literal_i$  the value  $FALSE$  iff  $P$  traverses through  $literal_i^j$ . Since the formula is not satisfied by this truth assignment, there is a clause vertex  $C_p^j$  such that all of its neighbors are in  $P$ . Then,  $l(P_{root,C_p^j}^T) \geq 1 - \frac{1}{k}$ , and  $r(P_{root,C_p^j}^T) = 1$ . Therefore,  $rad_t(T) \geq 2 - \frac{1}{k}$ .  $\blacksquare$

## 4 The quickest diameter spanning tree problem

In this section we consider the QUICKEST DIAMETER SPANNING TREE PROBLEM. We present a  $\frac{3}{2}$ -approximation algorithm, and prove that unless  $P = NP$  this is the **best possible**.

Denote by  $OPT$  the cost of an optimal solution,  $T_{opt} = (V, E_{opt})$ , to the QUICKEST DIAMETER SPANNING TREE PROBLEM. Denote by  $MDT(G, l)$  the minimum diameter spanning tree of a graph  $G$  where the edges are endowed with a length function  $l$ .

Algorithm *quick\_diameter*:

1. For every  $r \in \{r(e) : e \in E\}$  do
  - (a) Let  $G_r = (V, E_r)$ , where  $E_r = \{e \in E : r(e) \leq r\}$ .
  - (b)  $T'_r = MDT(G_r, l)$ .
2. Let  $T'$  be a minimum cost solution among  $\{T'_r\}_{r \in \{r(e) : e \in E\}}$ .
3. **Return**  $T'$ .

**Lemma 5** Let  $r_{max} = \max_{e \in E_{opt}} r(e)$ . For every pair of vertices  $u, v \in V$ ,  $\frac{l(P_{u,v}^{T_{opt}})}{2} + r_{max} \leq OPT$ .

**Proof:** The claim clearly holds if the path  $P_{u,v}^{T_{opt}}$  contains an edge  $e$  with  $r(e) = r_{max}$ . Assume otherwise (that the claim does not hold), and let  $y \in P_{u,v}^{T_{opt}}$  be such that there exists  $z \notin P_{u,v}^{T_{opt}}$ ,  $P_{y,z}^{T_{opt}}$  contains an edge  $e$  with  $r(e) = r_{max}$ , and  $P_{u,v}^{T_{opt}}$  and  $P_{y,z}^{T_{opt}}$  are edge-disjoint. Then,  $2OPT \geq t(P_{u,z}^{T_{opt}}) + t(P_{v,z}^{T_{opt}}) \geq l(P_{u,y}^{T_{opt}}) + l(P_{v,y}^{T_{opt}}) + 2r_{max} = l(P_{u,v}^{T_{opt}}) + 2r_{max}$ , and the claim follows. ■

For a tree  $T$ , denote by  $diam_l(T) = \max_{u,v \in V} l(P_{u,v}^T)$ .

**Theorem 6** Algorithm *quick\_diameter* is a  $\frac{3}{2}$ -approximation algorithm for the QUICKEST DIAMETER SPANNING TREE PROBLEM.

**Proof:** Algorithm *quick\_diameter* is polynomial, and returns a feasible solution. It remains to show the approximation ratio of the algorithm.

1. By the optimality of  $T'_{r_{max}}$  (for the minimum diameter spanning tree problem),  $diam_l(T'_{r_{max}}) \leq diam_l(T_{opt})$ .
2. By Lemma 5,  $diam_l(T_{opt}) \leq 2OPT - 2r_{max}$ . Therefore, by 1,  $diam_l(T'_{r_{max}}) \leq 2OPT - 2r_{max}$ .

3. Since  $T'_{r_{max}} \in G_{r_{max}}$ ,  $r(T'_{r_{max}}) \leq r_{max}$ .
4. By 2 and 3,  $diam_t(T'_{r_{max}}) = diam_l(T'_{r_{max}}) + r(T'_{r_{max}}) \leq 2OPT - r_{max}$ .
5. By 1,  $diam_l(T'_{r_{max}}) \leq diam_l(T_{opt}) \leq OPT$ .
6. By 5,  $diam_t(T'_{r_{max}}) \leq OPT + r_{max}$ .
7. By 4 and 6  $diam_t(T'_{r_{max}}) \leq \frac{3}{2}OPT$ . Therefore,  $diam_t(T') \leq \frac{3}{2}OPT$ .

■

**Theorem 7** *If  $P \neq NP$ , then there is no  $(\frac{3}{2} - \epsilon)$ -approximation algorithm for the QUICKEST DIAMETER SPANNING TREE PROBLEM for any  $\epsilon > 0$ .*

**Proof:** We prove the claim via reduction from SAT. We take the graph  $G$  from the proof of Theorem 4, and add a new vertex  $leaf$ , that is adjacent only to  $root$  by an edge  $e$  with  $l(e) = 1$  and  $r(e) = 0$ .

By the proof of Theorem 4, if the formula can be satisfied, then there is a tree,  $T$ , whose radius is 1 (by extending the tree derived in the proof of Theorem 4 with the edge  $(root, leaf)$ ). The diameter of a tree is at most twice its radius. Therefore, there is a spanning tree  $T$  such that  $diam_t(T) \leq 2$ .

Since  $leaf$  is adjacent (in  $G$ ) only to  $root$ , each feasible solution is decomposed into a spanning tree over the original vertex set and the edge  $(root, leaf)$ . By the proof of Theorem 4, if the formula cannot be satisfied, then every spanning trees  $T$  has a vertex  $u$  ( $u \neq leaf$ ) such that  $l(P_{root,u}^T) \geq 1 - \frac{1}{k}$  and  $r(P_{root,u}^T) \geq 1$ . Therefore,  $t(P_{u,leaf}^T) = l(P_{root,u}^T) + 1 + r(P_{u,leaf}^T) \geq 1 - \frac{1}{k} + 1 + 1 = 3 - \frac{1}{k}$ .

Given  $\epsilon > 0$  we can pick an integer  $k$  such that  $\frac{3 - \frac{1}{k}}{2} \geq \frac{3}{2} - \epsilon$ . ■

## 5 Discussion

In this paper we consider a pair of new problems that consider the transmission time along paths instead of the usual length of the paths. The two problems that we consider are to minimize the diameter and the radius of a spanning tree with respect to the transmission time. There are numerous other graph problems of interest that ask to compute a minimum cost subgraph where the cost is a function of the distances among vertices on the subgraph. Defining these problems under the transmission time definition of length opens an interesting area for future research.

## References

- [1] G.-H. Chen and Y.-C. Hung, "On the quickest path problem," *Information Processing Letters*, **46**, 125-128, 1993.
- [2] Y. L. Chen, "An algorithm for finding the  $k$  quickest paths in a network," *Computers & Opns. Res.*, **20**, 59-65, 1993.
- [3] Y. L. Chen, "Finding the  $k$  quickest simplest paths in a network," *Information Processing Letters*, **50**, 89-92, 1994.
- [4] Y. L. Chen and Y. H. Chin, "The quickest path problem," *Computers & Opns. Res.*, **17**, 153-161, 1990.
- [5] R. Hassin and A. Tamir, "On the minimum diameter spanning tree problem," *Information Processing Letters*, **53**, 109-111, 1995.
- [6] D. T. Lee and E. Papadopoulou, "The all-pairs quickest path problem," *Information Processing Letters*, **45**, 261-267, 1993.
- [7] E. Q. V. Martins and J. L. E. Santos, "An algorithm for the quickest path problem," *Operations Research Letters*, **20**, 195-198, 1997.
- [8] M. H. Moore, "On the fastest route to convoy-type traffic in flowrate-constrained networks," *Transportation Science*, **10**, 113-124, 1976.
- [9] B. Pelegrin and P. Fernandez, "On the sum-max bicriterion path problem," *Computers & Opns. Res.*, **25**, 1043-1054, 1998.
- [10] N. S. V. Rao, "Probabilistic quickest path algorithm," *Theoretical Computer Science*, **312**, 189-201, 2004.
- [11] J. B. Rosen, S. Z. Sun and G. L. Xue, "Algorithms for the quickest path problem and enumeration of quickest paths," *Computers & Opns. Res.*, **18**, 579-584, 1991.