JOURNAL OF ALGORITHMS **24,** 266–286 (1997)
ARTICLE NO. AL960848

Approximation Algorithms for Min–Max Tree Partition

Nili Guttmann-Beck and Refael Hassin*

Department of Statistics and Operations Research, Tel Aviv University, Tel Aviv, 69978, Israel

Received May 11, 1995

We consider the problem of partitioning the node set of a graph into p equal sized subsets. The objective is to minimize the maximum length, over these subsets, of a minimum spanning tree. We show that no polynomial algorithm with bounded error ratio can be given for the problem unless P = NP. We present an $O(n^2)$ time algorithm for the problem, where n is the number of nodes in the graph. Assuming that the edge lengths satisfy the triangle inequality, its error ratio is at most $2p - 1$. We also present an improved algorithm that obtains as an input a positive integer x. It runs in $O(2^{(p+x)p}n^2)$ time, and its error ratio is at most $(2 - x/(x + p - 1))p$.

1. INTRODUCTION

In the *min-max tree partition problem*, a complete weighted undirected graph $G = (V, E)$ is given, where V is its node set and E is the edge set, together with nonnegative edge lengths satisfying the triangle inequality. The set V must be partitioned into p equal-sized subsets. A minimum spanning tree (MST) is then found in each of the subgraphs induced by the partition. The objective is to minimize the weight of the longest MST.

The problem (as well as the related min-sum problem) is NP complete even for $p = 2$, as we prove in the appendix. We therefore develop approximation algorithms.

We present an $O(n^2)$ time algorithm whose error ratio is bounded by $2p - 1$, where $n = |V|$. We then describe an improved algorithmic scheme that gives a better bound, but with higher complexity. For any given value of a parameter $x \in \{1, 2, \ldots\}$ it runs in $O(2^{(p+x)p}n^2)$ time and its error ratio is bounded by $(2 - x/(x + p - 1))p$. For example, setting $x = \log n$, we obtain for any fixed p an $O(n^3)$ algorithm with an asymptotic error

* E-mail: {nili,hassin}@math.tau.ac.il.

0196-6774/97 $25.00

ratio of p. For p fixed and setting x to any slowly increasing unbounded function, the same asymptotic bound can be achieved in about $O(n^2)$ time (setting $x = \log\log n$ yields $O(n^2 \log n)$ time).

Goemans and Williamson in [4, 5] (see also [1, 7]), and Guttmann and Hassin in [3] gave approximate algorithms for partitioning G to achieve minimum *total* length of the MSTs in the partition. Let Σ^* be an optimal partition in the min-sum problem. Define $\mathrm{Max}(\Sigma^*)$ to be the length of the longest MST in an Σ^* and let $\mathrm{Sum}(\Sigma^*)$ be the sum of the lengths of all the trees in this solution. Let OPT be an optimal partition in the min-max problem; define Max(OPT) and Sum(OPT) in a similar way. Then

$$\mathrm{Max}(\Sigma^*) \le \mathrm{Sum}(\Sigma^*) \le \mathrm{Sum}(\mathrm{OPT}) \le p\,\mathrm{Max}(\mathrm{OPT}).$$

Similarly, an approximation algorithm with an error ratio at most α for the min-sum problem is also an approximation algorithm with an error ratio at most αp for the min-max problem.

For the min-sum version, Goemans and Williamson gave a bound of $4(1 - p/n)(1 - 1/n)$, which gives a bound of $4p$ for the min-max version and $p = o(n)$. In [3], for small values of p and any $\epsilon > 0$, a $2(1 + \epsilon)$-approximation was obtained for the min-sum problem, implying a $2p(1 + \epsilon)$-approximation for the min-max version. The present paper contains a $(1 + \epsilon)p$ approximation for the min-max version for every $\epsilon > 0$.

The idea of the approximation algorithm is as follows: Compute a MST on G. If the removal of some edges breaks it into two pieces whose sizes are multiples of n/p then do this and recurse; otherwise, double the edges to get a Hamiltonian cycle and break this cycle into p equal-sized pieces.

Our algorithms can also be used to approximate the problem of covering G by disjoint cycles. This can be done by doubling all the trees and using the triangle inequality to replace each tree by a cycle whose size is at most twice the size of the tree. The resulting error bound is twice the corresponding bound for the tree partition problem.

2. DEFINITIONS

For an edge e, $l(e)$ is the length of e.

For a set of edges $E' \subseteq E$, $l(E') = \sum_{e \in E'} l(e)$.

For a graph $G = (V, E)$, $l(G) = l(E)$.

For $V' \subset V$, MST(V') is a MST on the subgraph induced by V'.

For a subgraph B we denote by V_B and E_B the sets of nodes and edges in B, respectively.

Given a graph $G = (V, E)|V| = n$, where n is a multiple of p, the min-max tree partition problem (MMTP) is to partition V into disjoint sets P_i of size n/p each so that $\max_{1 \le i \le p}\{l(\mathrm{MST}(P_i)\}$ is minimized.

3. THE CYCLE PROCEDURE

The subject of this section is Procedure Cycle_Part, given in Fig. 1. However, we first present a general result.

Consider a cycle with edges of lengths $l_1, \ldots, l_n \geq 0$ (l_i is the length of edge e_i) in this (cyclic) order. For $i = 1, \ldots, n/p$ consider the partition of the cycle generated by deleting the edges with index $j = i \pmod{n/p}$. Let

Cycle_Part
input
1. A graph $G = (V, E)$, $|V| = n$.
2. A spanning tree T on G.
3. An integer p dividing n.
returns
$\{P_j\}_{j=0}^{p_0-1}$ *where* $\cup_{j=0}^{p_0-1} P_j = V_T$ *and* $|P_j| = \frac{|V_T|}{p_0}$.
begin
if $(p = 1)$
then
$P_0 := V_T$
return $\{P_0\}$
end if
Double all the edges in E_T. A cycle has been created.
Change the cycle into a simple cycle C of equal or smaller length, using the triangle inequality.
Number the nodes in V_T so that
$E_C = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)\}$.
for $(i = 1$ *to* $\frac{n}{p})$
$$r_i := \max_{0 \leq j \leq p-1} \sum_{a=i+j\frac{n}{p}}^{i+(j+1)\frac{n}{p}-2} l(v_a, v_{a+1}).$$
end for
Let i_0 be an index in $\{1, \ldots, \frac{n}{p}\}$ for which: $r_{i_0} = \min_{1 \leq i \leq \frac{n}{p}} \{r_i\}$.
for $(j = 0$ *to* $p - 1)$
$P_j := \{v_a | a = i_0 + j\frac{n}{p}, \ldots, i_0 + (j+1)\frac{n}{p} - 1\}$.
end for
return $\{P_j\}_{j=0}^{p-1}$
end *Cycle_Part*

FIG. 1. The cycle routines.

h^i denote the maximum length of a subpath generated by this partition:

$$h^i = \max_{j \in \{0, \ldots, p-1\}} \sum_{k=i+jn/p+1}^{i+(j+1)n/p-1} l_k.$$

LEMMA 3.1. *There exists* $i_0 \in \{1, \ldots, n/p\}$ *such that*

$$h^{i_0} \leq \frac{l(C)}{2}.$$

Proof. Suppose otherwise. Then, the removal of $e_i, e_{i+n/p}, \ldots$ creates on the cycle one path, S_0, satisfying $l(S_0) > l(C)/2$. Without loss of generality assume that

$$S_0 = \{e_{i+1}, e_{i+2}, \ldots, e_{i+n/p-1}\}.$$

The edges touching S_0 on both sides are e_i and $e_{i+n/p}$. The remove of the edges $e_{i+1}, e_{i+1+n/p}, \ldots$ again creates on the cycle one path, S_1, satisfying $S_1 > l(C)/2$. Since both $l(S_0)$ and $l(S_1)$ are greater than $l(C)/2$, it must be that $S_0 \cap S_1 \neq \phi$, so that

$$S_1 = \{e_{i+2}, e_{i+3}, \ldots, e_{i+n/p}\}.$$

If we continue in the same manner, defining S_j to be the longest path created on the cycle when removing $e_{i+j}, e_{i+j+n/p}, \ldots,$ then

$$S_j = \{e_{i+j+1}, e_{i+j+2}, \ldots, e_{i+j+n/p-1}\}.$$

In this case,

$$S_{n/p} = \{e_{i+j+n/p}, e_{i+j+n/p}+1, \ldots, e_{i+j+2n/p-1}\}.$$

So $S_0 \cap S_{n/p} = \phi$. But since $l(S_0) > l(C)/2$ and $l(S_{n/p}) > l(C)/2$, a contradiction. ∎

LEMMA 3.2. *Let* $\{P_i\}_{i=0}^{p-1}$ *be the partition returned by Cycle_Part. Then*

$$r = \max_{0 \leq i \leq p-1} l(MST(P_i)) \leq l(T).$$

Proof. Define $l_i = l(v_i, v_{i+1}) \geq 0$. According to Lemma 3.1 there is $i \in \{1, \ldots, n/p\}$ such that

$$h^i \leq \frac{l(C_r)}{2}.$$

From the way the cycle was created, $l(C_r) \le 2l(T)$. Therefore,

$$h^i \le l(T)$$

$$\Rightarrow \max_{0 \le j \le p-1} \sum_{k=i+jn/p+1}^{i+(j+1)n/p-1} l_k \le l(T)$$

$$\Rightarrow \sum_{k=i+jn/p+1}^{i+(j+1)n/p-1} l_k \le l(T) \qquad \forall j \in \{0, \ldots, p-1\}.$$

It follows from the definition of r_{i+1} that

$$r_{i+1} = \max_{0 \le j \le p-1} \sum_{a=i+1+jn/p}^{i+1+(j+1)n/p-2} l(v_a, v_{a+1}) \le l(T)$$

$$\Rightarrow r_{i_0} \le r_{i+1} \le l(T).$$

Since the edges $\{(v_{i_0+jn/p}, v_{i_0+jn/p+1}), \ldots, (v_{i_0+(j+1)n/p-2}, v_{i_0+(j+1)n/p-1})\}$ form a spanning tree of P_j,

$$l\big(\mathrm{MST}(P_j)\big) \le \sum_{k=i_0+jn/p}^{k=i_0+(j+1)n/p-2} l(v_k, v_{k+1}) \le r_{i_0}$$

$$\Rightarrow \max_{0 \le j \le p-1} l\big(\mathrm{MST}(P_j)\big) \le l(T)$$

$$\Rightarrow r \le l(T). \quad \blacksquare$$

To see that when $p \ge 3$ Cycle_Part may give a bad approximation consider the graph shown in Fig. 2a. There are three sets of two nodes each.

An edge between nodes inside the same set is of length 0. An edge connecting nodes from different sets is of length 1.

A MST for this graph is shown in Fig. 2b. Since $p = 3 \ne 1$ we double the edges to obtain the graph shown in Fig. 2c. The graph after the simple cycle is created is shown in Fig. 2d.

In this case

$$r_1 = r_2 = r_3 = 1.$$

i_0 can then be set to 2, giving $P_0 = \{v_2, v_3\}$, $P_1 = \{v_4, v_5\}$, $P_2 = \{v_6, v_1\}$. This partition is shown in Fig. 2e, giving a value $r = 1$, while an optimal partition with $opt = 0$ is shown in Fig. 2f.

When $p = 2$, *Cycle_Part* computes a bounded approximation. We will use the following theorem to prove it.

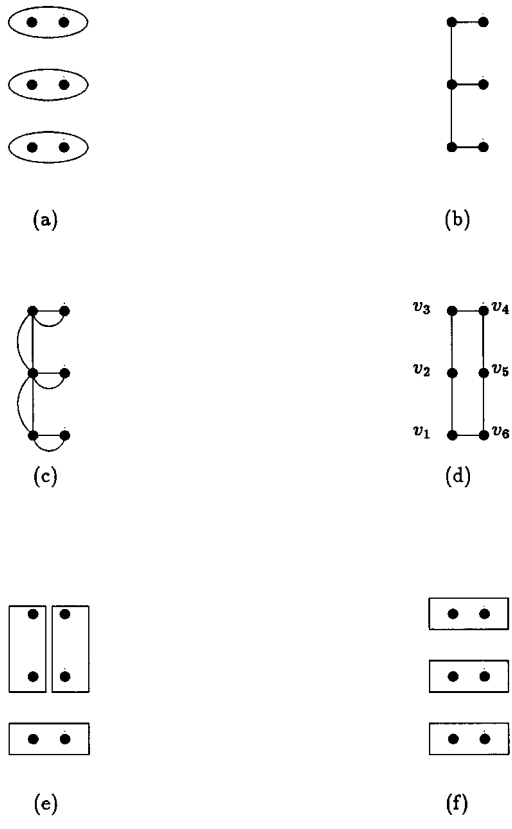

FIG. 2. A bad instance for Cycle_Part.

THEOREM 3.3 (Gale). *Given a graph $G = (V, E)$. Let $T_1 = (V, H)$ be a MST of G, and let $T_2 = (V, F)$ be a spanning tree of G. Suppose that*

$$H = \{h_1, h_2, \ldots, h_{n-1}\} \text{ is ordered so that } l(h_1) \leq \cdots \leq l(h_{n_1}),$$

$$F = \{f_1, f_2, \ldots, f_{n-1}\} \text{ is ordered so that } l(f_1) \leq \cdots \leq l(f_{n_1}).$$

Then

$$l(h_i) \leq l(f_i) \qquad \forall i \in \{1, \ldots, n-1\}.$$

The proof is given by Gale in [2] (see also [6]).

THEOREM 3.4. *When $p = 2$ and the input tree T_0 to Cycle_Part is a MST of G, the value r returned by this procedure satisfies*

$$r \leq 3\,\text{opt}.$$

Proof. Let O_1, O_2 be an optimal partition. Denote the set of edges of MST(O_1) and MST(O_2) as E_{O_1} and E_{O_2}, respectively. Let e^* be a shortest edge between O_1 and O_2:

$$l(e^*) = \min_{v \in O_1, u \in O_2} l(v, u).$$

According to the definitions $E_{O_1} \cup E_{O_2} \cup \{e^*\}$ is a spanning tree of G, hence:

$$\begin{aligned} l(T_0) &\le l(e^*) + l(\mathrm{MST}(O_1)) + l(\mathrm{MST}(O_2)) \\ &\le l(e^*) + 2\max\{l(\mathrm{MST}(O_1)), l(\mathrm{MST}(O_2))\}. \end{aligned}$$

Therefore,

$$l(T_0) \le l(e^*) + 2\,\mathrm{opt}.$$

There are two cases to be considered:

$l(e^*) \le$ opt. According to Lemma 3.2, the value r for the returned partition satisfies $r \le l(T_0)$. Therefore,

$$r \le l(e^*) + 2\,\mathrm{opt} \le 3\,\mathrm{opt}.$$

$l(e^*) >$ opt. In this case, the set of edges $E_{O_1} \cup E_{O_2} \cup \{e^*\}$ contains at most one edge of length $l(e^*)$. According to Theorem 3.3 T_0 contains at most one edge of length $l(e^*)$, so that T_0 contains at most one edge between O_1 and O_2. After doubling the edges there can be at most two edges between O_1 and O_2. Changing the cycle into a simple one does not change the number of edges between O_1 and O_2. Hence, the simple cycle C contains precisely two edges between O_1 and O_2. Since the number of nodes in O_1 and O_2 is equal, the cycle C is ordered to contain $n/2$ nodes from O_1, an edge from the O_1 to O_2, $n/2$ nodes from O_2, and a second edge between O_2 and O_2. Hence, in this case $r =$ opt. ∎

4. FIRST APPROXIMATION ALGORITHM

To partition G into p parts call Part_Alg(G, p), where Part_Alg is defined in Fig. 3. This algorithm uses the Cycle_Part defined in Fig. 1.

Step 1 of Part_Alg removes the longest edge of a MST of G. It then checks whether the size of each of the created components is a multiple of $|V|/p$. If the components satisfy this requirement, Part_Alg is recursively called to partition each of the components into parts of sizes $|V|/p$. Otherwise, Step 2 applies Cycle_Part to the MST.

Part_Alg
input
1. A graph $G = (V, E)$.
2. $T = MST(G)$.
3. An integer p *which divides* $|V|$.
returns
$\{P_j\}_{j=0}^{p-1}$ *where* $\cup_{i=0}^{p-1} P_j = V$ *and* $|P_j| = \frac{|V|}{p}$.
begin
Step 1 if $(p = 1)$
then return $\{V\}$.
end if
Find e_1 *the longest edge in* E_T *and remove it from* E_T.
Two connected components C_1 *and* C_2 *are created.*
if $(|V_{C_1}|$ *is a multiple of* $\frac{|V|}{p})$.
then
$a := \frac{|V_{C_1}|p}{|V|}$.
$G_1 :=$ *the subgraph of* G *induced by* V_{C_1}.
$T_1 :=$ *the subtree of* T *induced by* V_{C_1}.
$G_2 :=$ *the subgraph of* G *induced by* V_{C_2}.
$T_2 :=$ *the subtree of* T *induced by* V_{C_2}.
Call Part_Alg(G_1, T_1, a) *where:*
$\{P_0^*, \ldots, P_{a-1}^*\}$ *is the returned partitioning.*
Call Part_Alg$(G_2, T_2, p - a)$ *where:*
$\{P_a^*, \ldots, P_{p-1}^*\}$ *is the returned partitioning.*
return $\{P_0^*, \ldots, P_{p-1}^*\}$
end if
end Step 1
Step 2 *Call Cycle_Part*(T, p) *where:*
$\{P_0^*, \ldots, P_{p-1}^*\}$ *is the returned partitioning,*
return $\{P_0^*, \ldots, P_{p-1}^*\}$
end Step 2
end *Part_Alg*

FIG. 3. The partitioning algorithm.

4.1. *Evaluating Part_Alg*

LEMMA 4.1. *Let* $\{P_i\}_{i=0}^{p-1}$ *be the partition returned by Part_Alg. Then*

$$r = \max_{0 \le i \le p-1} l\big(\mathrm{MST}(P_i)\big) \le l(T).$$

Proof. The proof is by induction on p. For $p = 1$, the procedure returns V, giving that $r = l(\mathrm{MST}(V))$, and the lemma holds.

Assuming the hypothesis is correct for partitioning the graph into $p_0 < p$ sets, we prove its correctness for partitioning into p sets. We consider two cases.

- The partition returned by Part_Alg was found in Step 1. According to the induction's hypothesis (and since clearly $a < p$ and $p - a < p$), $r_1 = \max_{0 \le i \le a-1} l(\mathrm{MST}(P_i^*)) \le l(T_1)$ and $r_2 = \max_{a \le i \le p-1} l(\mathrm{MST}(P_i^*)) \le l(T_2)$.

  T_1 and T_2 are subtrees of T satisfying $l(T_1) \le l(T)$ and $l(T_2) \le l(T)$. Hence, $r_1 \le l(T)$ and $r_2 \le l(T)$, giving that $r \le l(T)$.

- The partition returned by Part_Alg was found in Step 2. In this case, the partition offered is the one returned from Cycle_Part(T, p). According to Lemma 3.2, $r \le l(T)$. ∎

Let $\{O_i\}_{i=0}^{p-1}$ be an optimal partition, and denote the set of edges of $\mathrm{MST}(O_i)$ as E_{O_i}, $i \in \{0, \ldots, p-1\}$. Thus,

$$\mathrm{opt} = \max_{0 \le i \le p-1} \big\{ l\big(E_{O_i}\big) \big\}.$$

For every $i \ne j \{i, j\} \subset \{0, \ldots, p-1\}$ define $e_{(i,j)}$ to be an edge connecting O_i and O_j such that

$$l(e_{(i,j)}) = \min_{v \in O_i,\, u \in O_j} \{ l(v, u) \}.$$

Define a graph G_0 where nodes represent the sets O_i, and the length of the edge between the node representing O_i and the node representing O_j is $l(e_{(i,j)})$ for all i and j.

Define $\{e_\alpha^*\}_{\alpha=1}^{p-1}$ to be the $p-1$ edges of a MST in this graph. Rename the edges thus: $l(e_1^*) \le l(e_2^*) \le l(e_3^*) \cdots \le l(e_{p-1}^*)$.

The set of edges $\bigcup_{i=0}^{p-1} E_{O_i} \cup \{e_1^*, \ldots, e_j^*\}$ defines a subgraph of G with $p - j$ connected components. Let $\{U_0^j, \ldots, U_{p-j-1}^j\}$ be the sets of nodes in these components. For $j = 0$, $\{U_0^0, \ldots, U_{p-1}^0\}$ is exactly $\{O_0, \ldots, O_{p-1}\}$.

LEMMA 4.2. *The shortest edge between a node in* U_i^j *and a node in* U_k^j *for* $i \ne k$, $\{i, k\} \subset \{0, \ldots, p-j-1\}$ *is of length* $\ge l(e_{j+1}^*)$.

Proof. The set of edges $\{e_1^*, \ldots, e_{p-1}^*\}$ is a MST in the graph G_0.

Suppose there is an edge g between a node in U_i^j and a node in U_k^j, such that $l(g) < l(e_{j+1}^*)$. Add a corresponding edge in G_0, $\hat{g}$, to $\{e_1^*, \ldots, e_{p-1}^*\}$. A cycle has been created (possibly consisting of two parallel edges). This cycle contains at least one edge, $\hat{f}$, from $\{e_{j+1}^*, \ldots, e_{p-1}^*\}$ (since $\{e_1^*, \ldots, e_j^*\}$ are all edges inside the U_i^j sets). Then, $l(\hat{f}) \geq l(e_{j+1}^*)$ and $\{e_1^*, \ldots, e_{p-1}^*\} \setminus \{\hat{f}\} \cup \{\hat{g}\}$ is a strictly shorter spanning tree then $\{e_1^*, \ldots, e_{p-1}^*\}$, contradicting the fact that the latter is a MST.

THEOREM 4.3. *Let $\{P_i\}_{i=0}^{p-1}$ be the partition returned by Procedure Part_Alg and let apx $= \max\{l(\mathrm{MST}(P_i)): 0 \leq i \leq p-1\}$. Then*

$$\mathrm{apx} \leq (2p-1)\mathrm{opt}.$$

Proof. Let T be a MST of G. $\bigcup_{i=0}^{p-1} E_{O_i} \cup \{e_1^*, \ldots, e_{p-1}^*\}$ is a spanning tree of G. Therefore,

$$\begin{aligned} l(T) &\leq \sum_{i=1}^{p-1} l(e_i^*) + \sum_{i=0}^{p-1} l(\mathrm{MST}(O_i)) \\ &\leq (p-1)l(e_{p-1}^*) + p \max_{0 \leq i \leq p-1} \{l(\mathrm{MST}(O_i))\} \\ &\leq (p-1)l(e_{p-1}^*) + p\,\mathrm{opt}. \end{aligned} \tag{1}$$

The rest of the proof is by induction on p: For $p = 1$, opt $= l(T)$, while the algorithm returns V, so that apx $=$ opt.

Assuming the hypothesis is correct for partitioning the graph into $p_0 < p$ sets, we prove its correctness for partitioning into p sets. We consider two cases:

1. opt $< l(e_{p-1}^*)$. Let q be the number of edges in $\{e_1^*, \ldots, e_{p-1}^*\}$ of length $l(e_{p-1}^*)$.

In this case, the set of edges $\bigcup_{i=0}^{p-1} E_{O_i} \cup \{e_1^*, \ldots, e_{p-1}^*\}$ is a spanning tree with at most q edges of length $\geq l(e_{p-1}^*)$. Then, according to Theorem 3.3, T contains at most q edges of this length. Removing from T its q longest edges will leave only edges of length $< l(e_{p-1}^*) = l(e_{p-q}^*)$. Consider $\{U_0^{p-q-1}, \ldots, U_q^{p-q-1}\}$, defined before. According to Lemma 4.2 a shortest edge between a node in U_i^{p-q-1} and a node in U_k^{p-q-1} for every i, k is at least as long as $l(e_{p-q}^*)$. Thus, after the removal of the q longest edges from T there are no edges left between nodes from different U^{p-q-1}s. T is disconnected into $q+1$ connected components, so that this partitioning has to be $\{U_0^{p-q-1}, \ldots, U_q^{p-q-1}\}$. So when removing the longest edge from T we remove an edge which connects nodes from two different sets among $\{U_0^{p-q-1}, \ldots, U_q^{p-q-1}\}$.

By the induction hypothesis,

$$r_1 \leq (2a-1)l(C_1).$$

Since C_1 is a subtree of T and since $a < p$:

$$r_1 < (2p - 1)l(T).$$

Similarly,

$$r_2 < (2p - 1)l(T),$$

giving that

$$r = \max\{r_1, r_2\} < (2p - 1)l(T).$$

2. $l(e^*_{p-1}) \leq \text{opt}$. From Lemma 4.1 and Eq. (1),

$$\text{apx} \leq l(T) \leq (p - 1)l(e^*_{p-1}) + p\,\text{opt} \leq (2p - 1)\text{opt}. \quad \blacksquare$$

4.2. *Complexity*

THEOREM 4.4. *The time complexity of Part_Alg is $O(n^2)$, where $n = |V|$.*

Proof. We first note that before calling Part_Alg we need to find a MST on G, and when leaving Part_Alg we need to find the length of the longest MST in the offered partition. Finding these MSTs takes $O(n^2)$, and should be added to the complexity of Part_Alg when the complexity of the approximation algorithm is evaluated.

We prove by induction on p that for some constant $C > 0$, Part_Alg requires at most $C(pn)$ time. Clearly, for $p = 1$ the inductive assumption holds.

Assuming the hypothesis is correct for partitioning the graph into $p_0 < p$ sets, we prove its correctness for partitioning into p sets.

If $|V_{C_1}|$ is not a multiple of n/p, Step 1 terminates in $O(n)$ time. Otherwise:

- Calling Part_Alg(G_1, T_1, a) takes (by the hypothesis) at most $Ca|V_{G_1}|$ time.
- Calling Part_Alg$(G_2, T_2, p - a)$ takes (by the hypothesis) at most $C(p - a)|V_{G_2}|$ time.

Since the function is convex, the worst case for the two calls to Part_Alg is when $|V_1| = n/p$, $|V_2| = n(1 - 1/p)$ and then they take $Cp(n/p + n(1 - 1/p) = Cp\beta n$, where $\beta < 1$. Altogether, Step 1 takes at most $O(Cpm)$. Procedure Cycle_Part requires as follows:

- $O(n)$ time to double the edges and find the simple cycle.
- To find i_0 we calculate for each i the length of the edges we remove from the cycle, and find the i for which the edges remove the longest length. This takes $O(n)$.

Thus, for large enough C, the computation time is bounded by Cpn and the dominating step is finding the MST in the start and end of the algorithm. Hence, the whole algorithm takes $O(n^2)$. ▮

4.3. *A Bad Example*

We now describe an instance such that Part_Alg$(G, 2)$ gives apx $= 3$ opt.

Consider the graph with four sets of nodes described in Fig. 4a. The distance between nodes in the same set is 0. The distance between nodes from different sets is 1. Let $p = 2$.

A MST T of the graph is shown in Fig. 4b, $l(T) = 3$.

Step 1 removes $\hat{e}$ and checks the size of the components created. Since one of them contains a single node, $|V_{C_1}|$ is not a multiple of $|V|/2 = 6$. The algorithm then continues to Step 2.

Step 2 calls Cycle_Part. $p = 2 \neq 1$ so we double the edges, yielding the graph shown in Fig. 4c. Changing the cycle into a simple one yields the graph in Fig. 4d. The numbering of the nodes is shown in this figure, and for the simplicity of the figure a node v_i is denoted just by its index i.

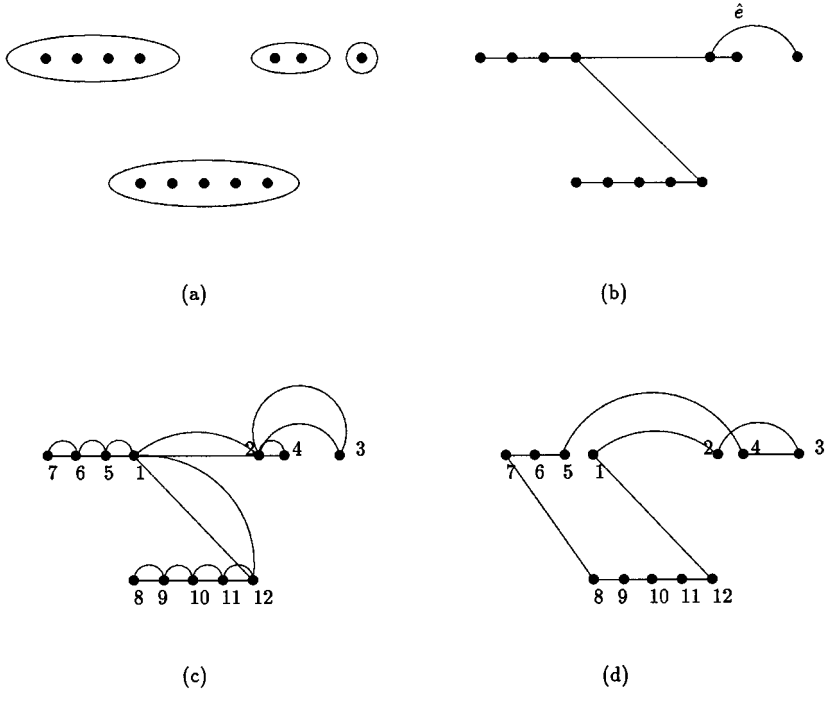

FIG. 4. A bad instance for Part_Algs.

According to the notation of the algorithm, $r_6 = 5$, $r_1 = r_5 = 4$, $r_2 = r_3 = r_4 = 3$, so i_0 may be any one of $\{2, 3, 4\}$. The algorithm may thus choose $i_0 = 4$. The offered partitioning is $(\{4, 5, 6, 7, 8, 9\}, \{10, 11, 12, 1, 2, 3\})$. *This partitioning is shown in Fig.* 5 *with* apx = 3. *An optimal partitioning is shown in Fig.* 5*b with* opt = 1. *Thus,* apx = 3 opt.

5. IMPROVING THE BOUND

In this section we present an algorithm with a better performance guarantee, at the expense of higher complexity. This algorithm defines a new parameter x that controls the improvement in the bound, and the higher complexity.

To partition G into p parts call Part_Alg_$x(G, p)$, defined in Fig. 6. This algorithm considers the $x + p - 1$ components obtained when $x + p - 2$ longest edges are removed from a MST of G. It considers all of the possible combination to aggregate part of these components into sets containing a multiple of $|V|/p$ nodes. For every such combination Part_Alg_x is recursively called to partition the above defined set of nodes and its complement. The combination which yields the best partitioning value is selected.

LEMMA 5.1. *Let* $\{P_i\}_{i=0}^{p-1}$ *be the partition returned by Part_Alg_x. Then*

$$r = \max_{0 \le i \le p-1} l(\mathrm{MST}(P_i)) \le l(T).$$

Proof. There are two cases to be considered:

- The partition returned by Part_Alg_x was found in Step 2.

In this case, $PT \neq \phi$, so that r calculated according to the partitioning PT must satisfy $r < l(T)$ (else it would not substitute for the previous value of r).

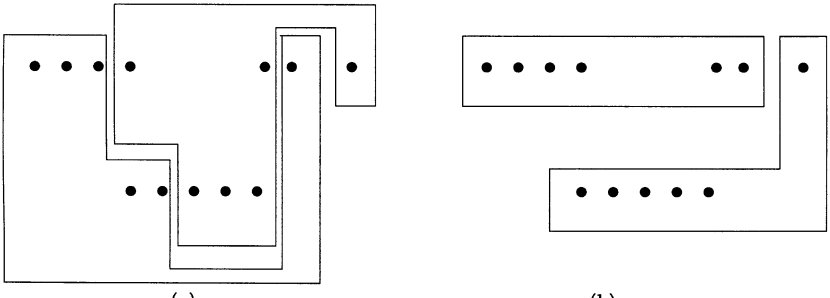

FIG. 5. The approximate (a) and optimal (b) solutions.

Part_Alg_x
input
1. A graph $G = (V, E)$.
2. An integer p *that divides* $|V|$.
returns
$\{P_j\}_{j=0}^{p-1}$ *where* $\bigcup_{j=0}^{p-1} P_j = V$ *and* $|P_j| = \frac{|V|}{p}$.
begin
Step 1
$T := MST(G)$
end Step 1
Step 2
if $(p = 1)$
then return $\{V\}$.
end if
$r := l(T)$. $PT := \phi$.
Remove the $x + p - 2$ *longest edges in* E_T.
A set of connected components $\{C_1, \ldots, C_{x+p-1}\}$ *is created.*
for every $(S \subset \{1, \ldots, x + p - 1\}$ *such that* $\frac{|V|}{p}$ *divides* $\sum_{j \in S} |V_{C_j}|)$
$V_1 := \cup_{j \in S} V_{C_j}$. $a := \frac{|V_1| p}{|V|}$.
$G_1 :=$ *The subgraph of* G *induced by* V_1.
$G_2 :=$ *The subgraph of* G *induced by* $V \backslash V_1$.
Call Part_Alg_x(G_1, a) *where:*
$\{P_0^*, \ldots, P_{a-1}^*\}$ *is the returned partitioning.*
Call Part_Alg_x$(G_2, p - a)$ *where:*
$\{P_a^*, \ldots, P_{p-1}^*\}$ *is the returned partitioning.*
$r^* = \max_{0 \le j \le p-1} l(MST(P_j^*))$.
if $(r^* < r)$
then $r := r^*$.
$PT := \{P_0^*, \ldots, P_{a-1}^*, P_a^*, \ldots, P_{p-1}^*\}$.
end if
end for
if $(PT \neq \phi)$
then
return PT
end if
end Step 2
Step 3 *Call Cycle_Part*(T, p) *where:*
$\{P_0^*, \ldots, P_{p-1}^*\}$ *is the returned partitioning.* **return** $\{P_0^*, \ldots, P_{p-1}^*\}$
end Step 3
end *Part_Alg_x*

FIG. 6. The improved partitioning algorithm.

• The partition returned by Part_Alg_x was found in Step 3. In this case, the partition was found by Cycle_Part(T, p). By Lemma 3.2, in this case too, $r \le l(T)$. ▮

THEOREM 5.2. *Let* $\{P_i\}_{i=0}^{p-1}$ *be the partition returned by Part_Alg_*$x(G, p)$. *Then*

$$\text{apx} = \max_{0 \le i \le p-1} l(\text{MST}(P_i)) \le \left(2 - \frac{x}{x+p-1}\right) p \text{ opt}.$$

Proof. Using the same definitions as in the proof of Theorem 4.3, it follows that Eq. (1) still applies. The rest of the proof is by induction on p. For $p = 1$, obviously apx $= l(T) =$ opt and since $2 - x/(x+p-1) = 1$ the proof is concluded.

Assuming the hypothesis is correct for partitioning the graph into $p_0 < p$ sets, we prove its correctness for partitioning into p sets. We consider two cases:

1. opt $< (x+p-1)/p) l(e^*_{p-1})$.

Let q be the number of edges in $\{e^*_1, \ldots, e^*_{p-1}\}$ of length $l(e^*_{p-1})$.

In this case, the number of edges of length $\ge l(e^*_{p-1})$ in a MST of a set in an optimal solution is less than or equal $(x+p-1)/p) - 1 = (x-1)/p$. The set of edges $\bigcup_{i=0}^{p-1} E_{O_i} \cup \{e^*_1, \ldots, e^*_{p-1}\}$ is a spanning tree of G with at most $p(x-1)/p + q = x - 1 + q$ edges of length $\ge l(e^*_{p-1}) = l(e^*_{p-q})$. Therefore, by Theorem 3.3, T will also contain at most $x - 1 + q$ edges of this length. Removing from T its $x - 1 + q$ longest edges will leave only edges of length $< l(e^*_{p-q})$. Consider $\{U_0^{p-q-1}, \ldots, U_q^{p-q-1}\}$ defined above. According to Lemma 4.2, a shortest edge between a node in U_i^{p-q-1} and a node in U_k^{p-q-1} for every i, k is at least as long as $l(e^*_{p-q}) = l(e^*_{p-1})$. Thus, after the removal of the $x - 1 + p - 1 \ge x - 1 + q$ longest edges from T there are no edges left between nodes from different U_i^{p-q-1}s. Hence, there is a subset $\{C_{i_1}, \ldots, C_{i_m}\}$ such that

$$\bigcup_{j \in S} V_{C_j} = U_0^{p-q-1}.$$

For this partitioning, according to the induction hypothesis (since $a < p$ and $p - a < p$),

$$\begin{aligned} r_1 &= \max_{0 \le i \le a-1} l(\text{MST}(P_i^*)) \\ &\le \left(2 - \frac{x}{x+a-1}\right) a \max_{O_i \subset U_0^{p-q-1}} l(\text{MST}(O_i)) \\ &\le \left(2 - \frac{x}{x+p-1}\right) p \max_{O_i \subset U_0^{p-q-1}} l(\text{MST}(O_i)) \end{aligned}$$

and

$$r_2 = \max_{a \le i \le p-1} l(\mathrm{MST}(P_i^*))$$
$$\le \left(2 - \frac{x}{x + (p-a) - 1}\right)(p-a) \max_{O_i \not\subset U_0^{p-q-1}} l(\mathrm{MST}(O_i))$$
$$\le \left(2 - \frac{x}{x+p-1}\right) p \max_{O_i \not\subset U_0^{p-q-1}} l(\mathrm{MST}(O_i)).$$

Therefore,

$$\max\{r_1, r_2\} \le \left(2 - \frac{x}{x+p-1}\right) p \max_{i=1,\ldots,p} l(\mathrm{MST}(O_i))$$
$$= \left(2 - \frac{x}{x+p-1}\right) p \,\mathrm{opt}.$$

According to the flow of the algorithm, at the end of the algorithm the value $r = \max_{0 \le i \le p-1} l(\mathrm{MST}(P_i))$ satisfies $r \le \max\{r_1, r_2\}$ for this partitioning. Therefore, the returned value $\mathrm{apx} = r$ satisfies

$$\mathrm{apx} \le \left(2 - \frac{x}{x+p-1}\right) p \,\mathrm{opt}.$$

2. $((x+p-1)/p) l(e_{p-1}^*) \le \mathrm{opt}$.

From Eq. (1),

$$l(T) \le (p-1) l(e_{p-1}^*) + p \,\mathrm{opt} \le \left(1 + \frac{p-1}{x+p-1}\right) p \,\mathrm{opt}$$
$$= \left(2 - \frac{x}{x+p-1}\right) p \,\mathrm{opt}.$$

Finally, from Lemma 5.1,

$$\mathrm{apx} \le l(T) \le \left(2 - \frac{x}{x+p-1}\right) p \,\mathrm{opt}.$$

THEOREM 5.3. *The time complexity of Part_Alg_x is* $O(2^{(p+x)p} n^2)$, *where* $n = |V|$.

Proof. As before, Cycle_Part takes $O(n)$. We use induction on p. For $p = 1$, the time is dominated by the MST computation, which is $O(n^2)$.

Suppose that for partitioning the graph into $p_0 < p$ sets the algorithm requires at most $C2^{(p_0+x)p_0}n^2$ time for some constant $C > 0$. Now consider partitioning into p sets.

Step 1 takes $O(n^2)$ time and Step 3 $O(n)$ time.

The time consuming operations of Step 2 consist of scanning the 2^{x+p} unions of components and whenever the number of nodes in the union is a multiple of $|V|/p$:

- Calling Part_Alg_$x(G_1, a)$ takes (by the hypothesis) at most $C2^{a(x+a)}|V_{G_1}|^2$.
- Calling Part_Alg_$x(G_2, p-a)$ takes (by the hypothesis) at most $C2^{(p-a)(x+p-a)}|V_{G_2}|^2$.

Altogether, this takes less than $2C2^{(p-1)(x+p-1)}n^2$. Multiplying by 2^{x+p-1} we obtain $2C2^{p(x+p-1)}n^2$, which for $p \geq 2$ is $\leq \frac{1}{2}C2^{p(x+p)}n^2$. Adding $C'n^2$ for Steps 1 and 3 and assuming $C \gg C'$, we obtain a bound of $C2^{p(x+p)}n^2$ as claimed. ▮

A COMPLEXITY OF THE PROBLEM

A.1. *NP-Completeness*

THEOREM A.1

The MMTP is NP-complete even for $p = 2$ and when the edge lengths satisfy the triangle inequality.

Proof. Consider the recognition version of the MMTP with $p = 2$: Given a graph $G = (V, E)$ and a constant K, find disjoint subsets $P, Q \subset V$ such that $|P| = |Q| = |V|/2$ and $l(\text{MST}(P)), l(\text{MST}(Q)) \leq K$.

It is easy to see that the problem is in NP. We will now reduce the satisfiability problem to MMTP via a polynomial transformation.

Given B, an instance of the satisfiability problem with variables $X_1, \ldots, X_n$ and clauses $C_1, \ldots, C_m$, we construct an instance of MMTP with $2(M^2 + mM + n)$ nodes (where $M = m + n$), and $K = M^2 + mM + n - 1$. Next we prove that this instance has P and Q as required if and only if B is satisfiable.

For each variable X_i, two nodes x_i and $\bar{x}_i$ are defined. For each clause C_j, M modes, $c_j^1, \ldots, c_j^M$, are defined. The reduction also adds sets of nodes $L = \{l_1, \ldots, l_{M^2}\}$, $E = \{e_1, \ldots, e_{M^2}\}$ and $D = \{d_1, \ldots, d_{mM}\}$.

The following pairs of nodes are connected by edges of length 1:

- x_i is connected to x_{i+1} and $\bar{x}_{i+1}$. $\bar{x}_i$ is connected to x_{i+1} and $\bar{x}_{i+1}$.
- $\forall i \in \{2, \ldots, mM\}$ d_i is connected to d_1.
- $\forall i \in \{1, \ldots, m\}$ $\forall j \in \{2, \ldots, M\}$ c_i^j is connected to c_i^1.
- $\forall j \in \{1, \ldots, m\}$ c_j^1 is connected to the node x_i (or $\bar{x}_i$) if X_i (or $\bar{X}_i$) is in the clause C_j.
- d_1 is connected to x_n and to $\bar{x}_n$.
- l_1 is connected to x_1, and to $\bar{x}_1$.
- e_1 is connected to c_1^1.
- $\forall i \in \{2, \ldots, M^2\}$ l_i is connected to l_1 and e_i is connected to c_1^1.

All the other edges (there is an edge between every two nodes) are of length 2.

The nodes of the graph and all the edges of length 1 for the expression

$$(x_1 + \bar{x}_2)(x_1 + x_3 + x_4)$$

where $m = 2$, $n = 4$, and $M = 6$, are described in Fig. 7.

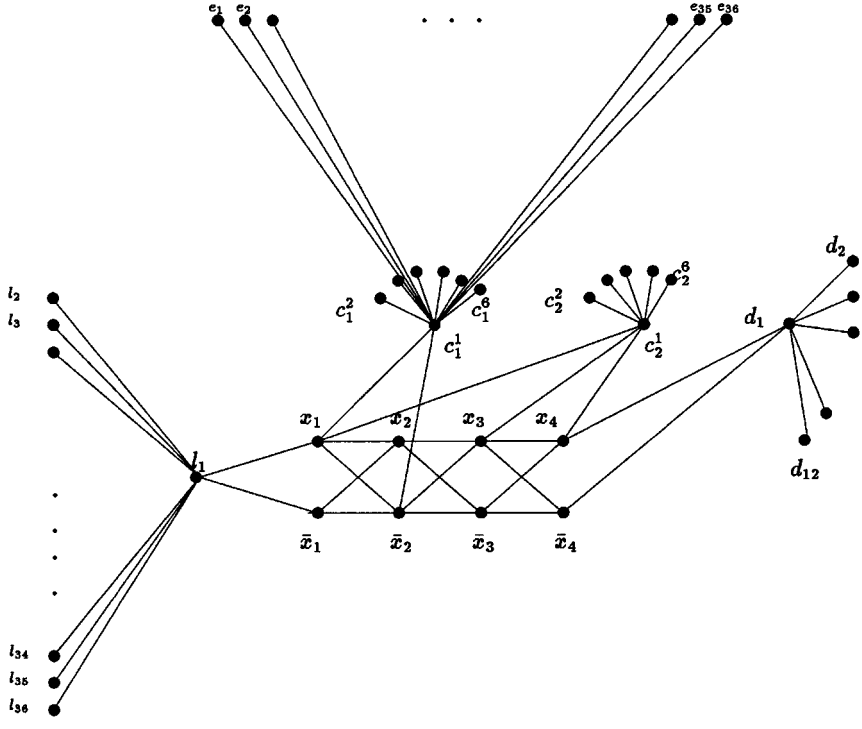

FIG. 7. The reduction of $(x_1 + \bar{x}_2)(x_1 + x_3 + x_4)$: $m = 2$, $n = 4$, $M = 6$.

The next step is to show that B is satisfiable if and only if the instance of MMTP has a solution.

Suppose the MMTP instance has a solution. In this case $|P| = |Q| = n + mM + M^2$. Since $K = n + mM + M^2 - 1$ and each of MST(P) or MST(Q) has exactly $n + mM + M^2 - 1$ edges, all the edges in MST(P) and MST(Q) must be of length 1.

Since P and Q have to be chosen so that all the edges of MST(P) and MST(Q) are of length 1, the next statements follow:

1. The nodes of D must be in the same set. Without loss of generality we assume that $D \subset P$.
2. $\forall i \in [1, \ldots, m\} \; \forall j \in \{2, \ldots, M\}$ c_i^j must be in the same set as c_i^1.
3. The nodes of L must be in the same set.
4. The nodes of E and c_1^1 must be in the same set.

Since $|L| + |E| = 2M^2 > |V|/2$ the sets L and E cannot be in the same set, P or Q.

We claim that $c_1^1 \notin P$. Suppose otherwise that $c_1^1 \in P$. From statement 2 it follows that $c_1^j \in P$ for every $1 \leq j \leq M$, and from statement 4 it follows that $E \subseteq P$. Altogether there are at least $mM + M + M^2$ nodes in P (mM nodes of D, M nodes that are the c_1^j nodes for all j, and M^2 nodes of E). But since M is greater than n, $mM + M + M^2 > mM + n + M^2 = |P|$, a contradiction. Therefore $c_1^1 \notin P$, and from statement 4, $E \not\subseteq P$. Consequently $L \subseteq P$. So far we have established that

$$L \subset P, \quad D \subset P, \quad E \subset Q,$$

and, for all j $c_1^j \in Q$.

Next we claim for all i, j $c_i^j \in Q$. Otherwise, by statement 2 there are at least M such nodes in P and with the nodes of D and L there are $mM + M^2 + M > |V|/2$ nodes in P, which is a contradiction.

So $L \cup D \subset P$, altogether $M^2 + Mm$ nodes, and there should be exactly n more nodes in P. The only way that the nodes in D can be connected to the nodes in L is by using a path of length $(n + 2)$ nodes, which starts at d_1, ends at l_1 and traverses on the way exactly n of the nodes from $\{x_i, \bar{x}_i\}_{i=1}^n$.

Let us name this path P_a. For every $i \in \{1, \ldots, n\}$ P_a contains either x_i or $\bar{x}_i$. Now for every i such that $x_i \in P_a$ we set $X_i =$ False. And for every i such that $\bar{x}_i \in P_a$ we set $X_i =$ True.

All that is left to show now is that this assignment satisfies B. All the nodes c_i^1 must be connected through a path of nodes in Q and edges of

length 1 to E. In particular, c_i^1 must be connected by an edge of length 1 to a node in Q which represents a literal of C_i. The complement node is in P_a, and the literal of C_i was therefore set to True.

We established that for every clause of B one of its literal was set to True and hence B is satisfiable as required.

On the other hand suppose that B is satisfiable. Let $P_a = (V_{P_a}, E_{P_a})$ be the path between d_1 and l_1 that traverses x_i if X_i is set to False and $\bar{x}_i$ if X_i is set to True.

Set $P = D \cup L \cup P_a$. P has exactly $m^2 + Mm + n$ nodes and clearly the MST of P contains only edges of length 1. Set $Q = C \cup E \cup V_{Q_a}$, where V_{Q_a} is defined to $\{x_i, \bar{x}_i\}_{i+1}^n \setminus V_{P_a}$.

Since B is satisfiable, for every clause C_j is at least one of its literals (x_i for example) is set to True. In this case $x_i \in Q_a$ and therefore in MST(Q) the node c_i^1 will be connected to x_i. Hence Q has exactly $m^2 + mM + n$ nodes, and MST(Q) contains only edges of length 1.

We established that P and Q are in the required size, and $l(\text{MST}(P)) = l(\text{MST}(Q)) = K$. ∎

A.2. *Approximability without the Triangle Inequality Assumption*

THEOREM A.2. *If P ≠ NP (and without assuming the triangle inequality), the MMTP problem has no polynomial approximation algorithm with bounded error guarantee, even when p = 2.*

Proof. Suppose to the contrary that there is a polynomial approximation algorithm and a constant $\alpha > 0$ such that for every instance B of the problem the algorithm finds a solution P_B, Q_B satisfying $\max\{l(\text{MST}(P_B)), l(\text{MST}(Q_B))\} \le \alpha \max\{l(\text{MST}(P_0)), l(\text{MST}(Q_0))\}$, where P_0, Q_0 is an optimal partition.

Given an instance B of the satisfiability problem, we construct the same graph as in the proof of Theorem A.1, except that in this case all the edges whose lengths were not set to 1 are now set to length (at least) $\alpha(M^2 + Mm + n - 1) + 1$. It follows from the above proof that B is satisfiable if and only if there exists a partition P, Q with MST(P) and MST(Q) containing only edges of unit length, and then $\max\{l(\text{MST}(P)), l(\text{MST}(Q))\} = M^2 + Mm + n - 1$. It follows that the approximation algorithm will find such a partition whenever it exists; otherwise it will use an edge of length $\alpha(M^2 + Mm + n - 1)$, in contradiction to the definition of α if P ≠ NP. ∎

A.3. *Complexity of Min-Sum Tree Partition*

Given $G = (V, E)$, $|V| = n$, n a multiple of p, the Min-Sum Tree Partition problem (MSTP) is to partition V into disjoint sets P_i, $|P_i| = n/p$, so that $\sum_{i=1}^{p} \{l(\text{MST}(P_i)\}$ is minimized.

THEOREM A.3. *If $P \neq NP$ (and without assuming the triangle inequality), the MSTP problem has no polynomial approximation algorithm with bounded error guarantee, even when $p = 2$.*

Proof. Again, consider the recognition version of the MSTP with $p = 2$, $k_1 = k_2 = |V|/2$. Given a graph $G = (V, E)$ and a constant K, find disjoint subsets $P, Q \subset V$ such that $|P| = |Q| = |V|/2$ and $l(\text{MST}(P)) + L(\text{MST}(Q)) \leq K$.

Again we build a reduction from the satisfiability problem as in the proof for Theorem A.2, only in this case set $K = 2(M^2 + mM + n - 1)$. The same proof as before will give the desired result. ▮

ACKNOWLEDGMENT

We thank the referees for their helpful comments. In particular, one of the referees suggested the proof of Lemma 3.1, which is much shorter than our original proof.

REFERENCES

1. H. N. Gabow, M. X. Goemans, and D. P. Williamson, An efficient approximation algorithm for the survivable network design problem, *in* "Proceedings of the Third MPS Conference on Integer Programming and Combinatorial Optimization," pp. 57–74, 1993.
2. D. Gale, Optimal assignments in an ordered set: An application of matroid theory, *J. Combin. Theory* **4** (1968), 176–180.
3. N. Guttmann and R. Hassin, "Approximation Algorithms for Minimum Tree Partition," Tel Aviv University, Tel Aviv, 1995.
4. M. X. Goemans and D. P. Williamson, A general approximation technique for constrained forest problems, *SIAM J. Comput.* **24** (1995), 296–317.
5. M. X. Goemans, and D. P. Williamson, Approximating minimum-cost graph problems with spanning tree edges, *Oper. Res. Lett.* **16** (1994), 183–189.
6. E. L. Lawler, "Combinatorial Optimization: Networks and Matroids," Holt, Rinehart & Winston, New York, 1976.
7. D. P. Williamson, "On the Design of Approximation Algorithms for a Class of Graph Problems," Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1990.