Applications of the Matroid Parity Problem to Approximating Steiner Trees

Piotr Berman* Martin Fürer† Alexander Zelikovsky‡

Abstract

The Steiner tree problem in graphs requires to find a minimum size connected subgraph containing a given subset of nodes (given points). In this paper we consider this problem in three classes of graphs: where the maximum path distance is 2, where given points form a dominating set and where the given points form a vertex cover. As all these problems are MAX-SNP hard, the issue is what approximation can be obtained in polynomial time. In the first case we obtain an approximation ratio (of the achieved size over the minimal one) $\frac{5}{4}+\frac{3}{80}=1.2875$, in the second case we achieve 4/3, and in the last case we achieve $8/7-1/160$.

1 Introduction

One of the strongest results in matroid theory is the polynomial time solution of the parity problem in linear matroids. Briefly stated, we have a collection of $2 \times n$ matrices, and our goal is to find a maximum size sub-collection such that they can be stacked into a single matrix of maximum rank. This problem was solved by Lovász [4], and much more efficient algorithms were later found by Orlin and Vande Vate [5] and Gabow and Stallmann [3] for graphic matroids.

Numerous graph problems can be expressed in the language of linear matroids, therefore one should expect that a number of them would be solved using an algorithm for matroid parity. Known examples are finding a minimum feedback vertex set in a cubic graphs, equivalently, a maximum size acyclic induced subgraph and finding a maximum planar subgraph [2]. Another is finding a Steiner minimum tree in a graph where the given points form an independent set and a vertex cover and the non-given points are adjacent to at most three given points each. However, until now, very few such applications were found. We hope that a technique described in this paper will allow to find many more such applications.

We will show approximation algorithms for three MAX-SNP hard versions of the Steiner tree problem. In each, we are given a graph with a set P of n given points, and the goal is to find a subset of edges that forms a tree containing all the given points. The cost of the tree is defined as the number of edges it contains. We measure the quality of our algorithm by the *performance ratio*, i.e. the worst case ratio of the cost of the obtained solution to the cost of an optimum solution.

In the first problem we restrict ourselves to graphs where every two nodes have a distance 1 or 2. Bern and Plassman [1] have already investigated this problem from an approximation point of view. They have shown that Rayward-Smith's heuristic [6] has a performance ratio 4/3 for such graphs. We generalize Rayward-Smith's heuristic and combine it with the exact algorithm for the matroid parity problem. We prove that the performance ratio of the combined algorithm is at most 5/4+3/80.

In the other two problems we restrict ourselves to graphs where P forms a dominating set or a vertex cover respectively. We will show how to reduce the problem of finding approximate solutions to approximating the *base cover problem*, which in turn we accomplish using the algorithm for matroid parity.

*Department of Computer Science and Engineering, Pennsylvania State University, E-mail: berman@cse.psu.edu
†Department of Computer Science and Engineering, Pennsylvania State University, E-mail: furer@cse.psu.edu
‡Computer Science Department, University of California at Los Angeles, E-mail: alexz@cs.virginia.edu. Supported in part by the Packard Foundation and Volkswagen Stiftung.

In this paper we use the following notation and terminology:

Terminology:

P is the set of given points.

$\#A$ is the cardinality of A.

$N(a, B)$ is the set of nodes of B that are adjacent to a.

If A is a set of objects and a is an object, $A + a$ and $A - a$ are shorthands for $A \cup \{a\}$ and $A - \{a\}$.

In a graph, collapsing a set of nodes means viewing them as a new single node. Afterwards, internal edges of the set are disregarded, and edges incident on any element of the set will be viewed as incident on the new node.

A *Steiner tree* is a tree (a connected acyclic subgraph) containing P, the set of *given points*. This tree is a union of subtrees in which no given point is an internal node and all leaves are given points; we call such subtrees *full components*. An internal node of a Steiner tree is called a *Steiner point* if its degree is at least 3.

For a graph we consider the following linear space: every set of nodes is a vector, and the field is $\mathbf{Z}_2$, thus the addition operation is the symmetric difference. This way, the edges of the graph, being two-element sets, are vectors in this linear space, and the singleton sets form a basis.

2 Graphs with Edge Lengths 1 and 2

In this section we consider the case of the Steiner tree problem in a complete graph G with edge lengths 1 and 2. This problem was first considered by Bern and Plassmann in [1] where they proved the MAX SNP hardness of this particular case of the Steiner tree problem and, therefore, the MAX SNP hardness of the general problem. They also suggest the 4/3-approximation algorithm which we improve in this section.

Let P ($\#P = n$) be the set of given points. Two vertices in distance 1 are called *neighbors*. A *frame* is a full tree with edge lengths 1, i.e. it is a tree in G with edge lengths 1, leaves from P and internal vertices from $V - P$. An *i-star* Z_i is a frame with the unique Steiner point (a *center*) $v(Z_i)$ adjacent to i given points neighboring $v(Z_i)$. Consider a subgraph $G(P)$ induced by the set of given points. Note that 3-stars correspond to pairs of vectors in the linear space for $G(P)$. Therefore, using Lovász' algorithm we can find the largest possible number $t_3(P)$ of *independent* 3-stars, i.e. 3-stars with an acyclic union. We will also use frames that are not i-stars (Fig. 1). The key operation of our algorithm (as well as Rayward-Smith's) is *contracting* of a frame. To *contract* a frame means to collapse all its given points.

Algorithm 1:

(* Quick stage *)
Sequentially contract all i-stars ($i = \ldots, 6, 5, 4$)
(* Exact stage *)
Contract the maximum independent set of 3-stars.
(* Frame stage *)
Sequentially contract all frames from F_1 and F_2.
(* Final stage *)
Output the minimum spanning tree (MST) of the given point set
with the Steiner points of all contracted frames.

Note that Rayward-Smith's heuristic from [1] consists of the quick stage for $i = ...6, 5, 4, 3$ and the final stage.

First, we consider the quick stage when $i \geq 5$. Each contraction of an i-star decrements the lengths of the both: the approximate and optimal solutions. The ratio between these two decrements is at most 5/4. Therefore, for any $\alpha \geq \frac{5}{4}$, an α-approximation for the instances without 5-stars gives an α-approximation for all instances. So from now on we may assume that there are no i-stars for $i \geq 5$.

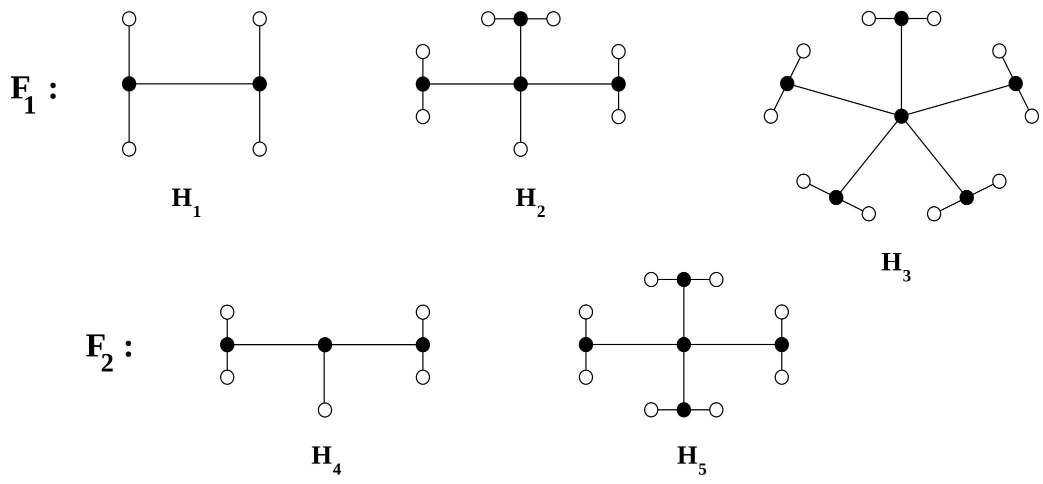

Figure 1: Two sets of frames $F_1 = \{H_1, H_2, H_3\}$ and $F_2 = \{H_4, H_5\}$. Steiner points and given points are denoted by black and white circles, respectively.

Similarly we may contract all pairs of neighboring given points. So we may assume that the cost of a minimum spanning tree of P, $mst = mst(P)$, equals $2n - 2$. Let OPT be an optimal Steiner tree for P of cost *opt*. The *Steiner ratio* is the ratio of mst over *opt*.

Define a partition of P by assigning each given point to a neighboring Steiner point of OPT. If a given point has two or more neighbors in OPT, then we arbitrary assign it to one of them. Let c_i be the number of equivalence classes with i given points. Thus, OPT contains c_i pairwise disjoint i-stars, $i = 3, 4$, and $c_i = 0$ for all $i \geq 5$.

Lemma 1 *Let t_4 and t_3 be the number of 4-stars and 3-stars chosen in the quick and exact stage, respectively. Then*

$$t_3 + 2t_4 \geq c_3 + 2c_4$$

Proof. Let B be the set of pairwise disjoint c_3 3-stars and c_4 4-stars from OPT. Let B_3 and A be the families of 3-stars and 4-stars chosen by Algorithm 1, respectively. After each greedy choice of a 4-star (Quick Stage, $i = 4$), the size of the family A (initially empty) is incremented by 1, while the family B of 3-stars and 4-stars (finally B_3) is modified.

Let $G(A \cup B)$ be the subgraph induced by $A \cup B$ and let K be a connected component of $G(A \cup B)$. Let K consist of a stars from A and k_4 4-stars and k_3 3-stars from B. For the component K we define the *A-potential* to be $p_A(K) = 2a$, the *raw B-potential* to be $r_B(K) = k_3 + 2k_4$, the *B-potential* to be $p_B(K) = \lceil \frac{2}{3} r_B(K) \rceil$, and the *potential* to be $p(K) = p_A(K) + p_B(K)$. The potential $p(A \cup B)$ of the entire graph $G(A \cup B)$ is the sum of the potentials of its components.

To prove the lemma, it suffices to verify the following three facts:

(i) At the beginning, when A is empty and B is unchanged, the potential is $c_3 + 2c_4$.

(ii) After each insertion of a 4-star into A, we can modify B in such a way that the potential does not decrease.

(iii) Finally, when B does not contain any 4-stars, the number of 3-stars c_3 is at least the B-potential of $A \cup B$.

The fact (i) is true, since at the beginning $G(A \cup B)$ is a union of pairwise disjoint 3-stars and 4-stars. The fact (iii) holds because the B-potential cannot be more than the number of 3-stars in B

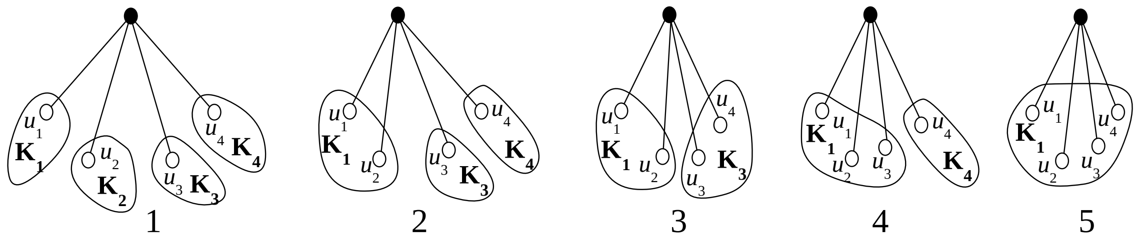

Figure 2: Five cases of insertion a 4-star to the set $A \cup B$.

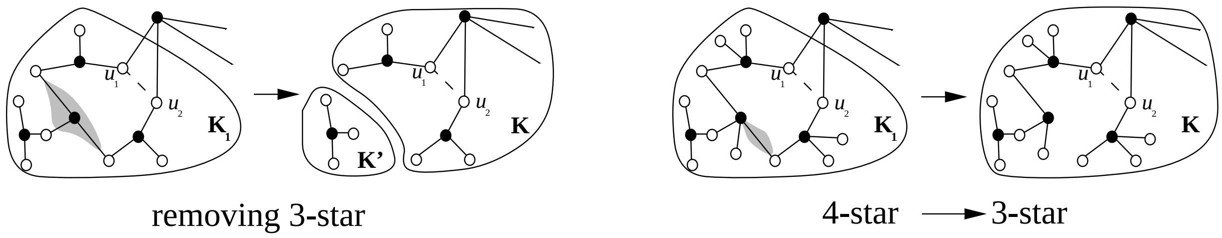

Figure 3: Breaking a cycle by removing a 3-star or by replacing a 4-star with a 3-star.

which in turn is at most the number of independent 3-stars that can be extracted by Lovász' algorithm in the exact stage of Algorithm 1. We finish the proof of Lemma 1 with

Proof of (ii). We need some more notations. We denote the terminals of a 4-star F that is being inserted into A by u_1, u_2, u_3, and u_4. Let K_i be the connected component of $G(A \cup B)$ containing u_i. For brevity, $r_i = r_B(K_i)$, $i = 1, 2, 3, 4$. Let $t = \sum^*(r_i \div 3)$, where the sum $\sum^*$ is taken over all *different* components among K_i, $i = 1, 2, 3, 4$. We denote by K the newly created connected component containing the 4-star F.

The insertion of F increases the A-potential by 2. We need to check that the B-potential decreases by at most 2 for all possible (mod 3)-remainders of r_i. Note that in the worst case $r_i \equiv 2 \pmod 3$, $i = 1, \ldots, 4$. Indeed, decrementing the remainder of r_i by 1 or 2 results in decrementing the original B-potential $\sum^* \lceil \frac{2}{3} r_i \rceil$ by 1 or 2, respectively. On the other hand, it results in decrementing of $\lceil \frac{2}{3} \sum^* r_i \rceil$ by *at most* 1 or 2, respectively. So furthermore, we assume that $r_i \equiv 2 \pmod 3$.

We need to consider the following five cases (Fig. 2):

Case 1. All components K_i are different. We need to check that $p_B(K) \geq \sum_{i=1}^4 p_B(K_i) - 2$. Indeed, then $p_B(K) = 6 + 2t$ and $\sum_{i=1}^4 p_B(K_i) = 8 + 2t$.

Case 2. $K_1 = K_2$. If the edge (u_1, u_2) forms a cycle with some 3-stars from K_1, then removal of one such 3-star from B (Fig. 3) breaks this cycle and partitions K_1 into 3 connected components. Let K' be the component resulted from breaking K_1 that does not contain either u_1 or u_3. Note that $r_B(K) + r_B(K') \geq r_1 + r_3 + r_4 - 1$. Therefore, $p_B(K) + p_B(K') = 4 + 2t$ while $\sum_{i=1,3,4} p_B(K_i) = 6 + 2t$.

If the cycle with (u_1, u_2) contains only 4-stars from K_1, then we can break the cycle replacing some 4-star from B with a 3-star (Fig. 3) and having $p_B(K) = 4 + 2t$.

Case 3. $K_1 = K_2$ and $K_3 = K_4$. This case is similar to Case 2. This time we have two cycles with edges (u_1, u_2) and (u_3, u_4) in components K_1 and K_3, respectively. In order to break these two cycles we remove 3-stars or replace 4-stars with 3-stars. Anyway, we decrease the raw B-potential by 2, making it equal to $r_1 + r_3 - 2$. Therefore, the updated B-potential equals $2 + 2t$ while the original one equals $4 + 2t$.

Case 4. $K_1 = K_2 = K_3$. It is quite similar to Case 3, only now we have two cycles with edges

(u_1, u_2) and (u_2, u_3) in the component K_1.

Case 5. $K_1 = K_2 = K_3 = K_4$. In the component K_1, we have 3 cycles containing three independent edges between u_i, $i = 1, 2, 3, 4$. This time, to break three cycles we need three times either to remove a 3-star or to replace a 4-star with a 3-star. Therefore, we decrease the raw B-potential by 3 which becomes equal to $r_1 - 3$. It is obvious that this decreases the B-potential by 2. □

Now we will bound the gain of the quick and exact stages $g_1 = mst(P) - mst(P \cup SP_{43})$ obtained by adding centers of 4-stars and 3-stars, SP_{43}. We assumed that no equivalence class contains more than 4 given points, so the tree OPT contains $n = 4c_4 + 3c_3 + 2c_2 + c_1$ given points and at least $c_4 + c_3 + c_2 + c_1$ Steiner points. Thus $g_1 = 2t_4 + t_3 \geq 2c_4 + c_3$ (Lemma 1) and

$$\begin{aligned}
\frac{mst - \frac{2}{3}g_1}{opt} &\leq \frac{8c_4 + 6c_3 + 4c_2 + 2c_1 - 2 - \frac{2}{3}g_1}{5c_4 + 4c_3 + 3c_2 + 2c_1 - 1} \\
&\leq \frac{\frac{20}{3}c_4 + \frac{16}{3}c_3 + 4c_2 + 2c_1 - 2}{5c_4 + 4c_3 + 3c_2 + 2c_1 - 1} \\
&\leq \frac{4}{3}
\end{aligned}$$

So we proved the following inequality

$$2g_1 \geq 3mst - 4opt \qquad (1)$$

Lemma 2 *Let a given point set C not have any i-stars for $i \geq 3$. If C does not contain frames from F_1, then $\frac{mst(C)}{opt(C)} \leq \frac{5}{4}$. If C does not contain frames from F_1 or F_2, then $\frac{mst(C)}{opt(C)} \leq \frac{6}{5}$.*

Proof. We denote by N_i the set of Steiner points of OPT adjacent to i given points and by n_i its size $\#N_i$, $i = 0, 1, 2$. Let C not contain any frames from F_1. Since frames H_2 and H_3 are forbidden, any N_1-point and N_0-point may be adjacent to at most 2 and 4 N_2-points, respectively. N_2 is an independent set since H_1 is forbidden. If an N_2-point v is adjacent only to two given points, then collapsing of this pair of given points and removal of v decreases the cost of both MST and OPT by 2, therefore it may only increase the Steiner ratio. Thus we may assume that $n_2 \leq 2n_1 + 4n_0$ and $n \leq 2n_2 + n_1$. So

$$\begin{aligned}
\frac{mst}{opt} &\leq \frac{4n_2 + 2n_1 - 2}{3n_2 + 2n_1 + n_0 - 1} \\
&\leq 1 + \frac{n_2 - n_0 - 1}{3n_2 + 2n_1 + n_0 - 1} \\
&\leq 1 + \frac{n_2 - n_0 - 1}{4n_2 - 3n_0 - 1} \\
&\leq \frac{5}{4}
\end{aligned}$$

Let C not contain any frames from F_1 or F_2. Since frames H_4 and H_5 are forbidden, any N_1-point and N_0-point may be adjacent to at most 1 and 3 N_2-points, respectively. Similarly, we obtain $n_2 \leq n_1 + 3n_0$ which yields that the Steiner ratio is at most $\frac{6}{5}$. □

Now we will bound the gains of the frame stage g_2 and g_3 obtained by adding Steiner points of the frames from F_1 and F_2, respectively, i.e.

$$g_2 = mst(P \cup SP_{43}) - mst(P \cup SP_{43} \cup SP_{F_1})$$

$$g_3 = mst(P \cup SP_{43} \cup SP_{F_1}) - mst(P \cup SP_{43} \cup SP_{F_1} \cup SP_{F_2})$$

The contraction of a frame Z decreases the cost of OPT by the number of covered base vectors, i.e. the dimension of Z in the linear space for $G(P)$. We denote this value by $\beta(Z) = \#(P \cap Z) - 1$. ¿From the other side, it decrease the cost of MST by $2\beta(Z)$. Let Z^* be the set of 4-and 3-stars chosen by the algorithm, and let F_1^* and F_2^* be the respective sets of F_1-frames and F_2-frames. Lemma 2 yields

$$\frac{mst - 2\beta(Z^*) - 2\beta(F_1^*)}{opt - \beta(Z^*) - \beta(F_1^*)} \leq \frac{5}{4} \tag{2}$$

$$\frac{mst - 2\beta(Z^*) - 2\beta(F_1^*) - 2\beta(F_2^*)}{opt - \beta(Z^*) - \beta(F_1^*) - \beta(F_2^*)} \leq \frac{6}{5} \tag{3}$$

Let $g(Z)$ be the gain of addition of Steiner points of a frame Z to the given point set. It is easy to check that for an i-star Z_i, $g(Z_i) = \beta(Z_i) - 1$, therefore, $g_1 = g(Z^*) \geq \frac{1}{2}\beta(Z^*)$. Moreover, for any $H \in F_j$, $g(H) \geq \frac{1}{2+j}\beta(H)$, $j = 1, 2$. Thus inequalities (2-3) yield

$$6g_1 + 9g_2 \geq 4mst - 5opt \tag{4}$$

$$8g_1 + 12g_2 + 16g_3 \geq 5mst - 6opt \tag{5}$$

Now we are ready to prove the main result of the section

Theorem 1 *The approximation ratio of Algorithm 1 for the Steiner tree problem in graphs with distances 1 and 2 is at most* $\frac{103}{80} \approx 1.2875$.

Proof. Let $appr$ be the cost of the output of Algorithm 1 and let $g = mst - appr$ be the total gain of Algorithm 1 with respect to the minimum spanning tree. The inequality $17\times(1)+1\times(4)+5\times(5)$ yields

$$80g_1 + 69g_2 + 80g_3 \geq 80mst - 103opt.$$

Thus the total gain $g = g_1 + g_2 + g_3$ is at least $mst - \frac{103}{80}opt$ and the performance ratio is

$$\frac{appr}{opt} = \frac{mst - g}{opt} \leq \frac{103}{80} = 1.2875.$$

□

3 The Base Cover Problem

We are given a collection C of sets of vectors from some n-dimensional linear space H. Our task is to find a minimum cost base of H, assuming that we can "purchase" a set of vectors d for the price of $\#d + 1$, provided that $d \subseteq c$ for some $c \in C$. More formally,

$\tilde{D} = \cup_{d \in D} d$
$span(D)$ = linear space spanned by $\tilde{D}$
$valid(D) \equiv D$ is a valid partial solution
$\equiv$ sets of D are pairwise disjoint and each set in D has a superset in C and $\tilde{D}$ is an independent set of vectors
$cost(D) = \sum_{d \in D}(\#d + 1) = \#\tilde{D} + \#D$

We will minimize $Cost(D)$ such that $valid(D)$ and $span(D) = H$. Of course, we need to assume that $span(C) = H$.

In two cases there exists a polynomial time exact algorithm. If C consists of singleton sets, every solution costs $2n$. If C is a collection of singletons and pairs, this is equivalent to the linear matroid parity problem, as we are maximizing the number of times when we can purchase two vectors for the price of 3 (thus minimizing the number of times when we purchase vectors individually, for the price of 2.)

To formulate our important observations, we need some more notation.

$D_i = \{d \in D | \#d \geq i\}$
$valid_i(D) \equiv valid(D)$ and $D = D_i$
$rank(D)$ = dimension of $span(D)$ = (if $valid(D)$) $\#\tilde{D}$
$cost_1(D) = \min\{cost(E) | rank(E) = n$ and $valid(E)$ and $E_2 = D\}$
(used if $valid_2(D)$, it is the cost of a solution obtained from D by adding singleton sets, equals $cost(D) + 2(n - rank(D))$
$rank_2(D) = \max\{rank(E) | rank(E) = n$ and $valid_2(E)$ and $E_3 = D\}$
(used if $valid_3(D)$, it is the largest rank of a valid solution obtained from D by inserting pairs)
$cost_2(D) = \min\{cost_1(E) | valid(E)$ and $E_3 = D\}$
(used if $valid_3(D)$, it is the least cost of a valid solution obtained from D by inserting pairs and singleton sets
$rank_3(D) = \max\{rank_2(E) | valid_3(E)$ and $D \subseteq E\}$
(used if $valid_3(D)$, it is the largest rank of a valid solution obtained from D by inserting non-singleton sets)
$r_{max} = \max\{rank(D) | valid_2(E)\} = rank_3(\emptyset)$
$c_{min} = \min\{cost(D) | valid(D)$
H/I is defined for a subspace I of the linear space H, it is a result of a linear mapping l such that $l^{-1}(0) = I$

Our algorithm requires that we can compute $rank(D)$ in polynomial time. If $valid_2(D)$, then $cost_1(D) = 2n - (\#\tilde{D} - \#D)$. Assuming $valid_3(D)$, we can find the largest collection D' of pairs of vectors such that $valid_2(D \cup D')$ by solving the linear matroid parity problem in $H/span(D)$; if $\#D' = k$, then $rank_2(D) = rank(D) + 2k$ and $cost_2(D) = cost_1(D) - k$. Thus to minimize $cost(D)$ such that $valid(D)$ it suffices to minimize $cost_2(D)$ such that $valid_3(D)$. The next two lemmas convey the crux of our method; the first lemma says that we can restrict our search to partial solutions satisfying $rank_3(D) = r_{max}$, the second lemma allows to compute $rank_3$ and to perform the analysis of our (almost) greedy algorithm.

Lemma 3 *Assume $valid_2(D)$ and $rank(D) = r_{max}$. Then there exists E such that $span(E) = span(D)$ and $cost_1(E) = c_{min}$.*

Proof. Let $conform(D, E)$ be the union of set intersections $d \cap e$ such that $d \in D$, $e \in E$ and $\#(d \cap e) \geq 2$. Among E such that $valid_2(E)$ and $cost_1(E) = c_{min}$ we can choose one that maximizes the size of $conform(D, E)$. It suffices to show that $span(D) \subseteq span(E)$: because $rank(D) = r_{max} \geq rank(E)$, $span(E)$ cannot properly contain $span(D)$, thus this inclusion implies equality.

Suppose, by the way of contradiction, that $span(D) \not\subseteq span(E)$. Then $\tilde{D} \not\subseteq span(E)$ and for some $u \in d \in D$ we have $u \notin span(E)$. Because $valid_2(D)$ holds, d has another element say v. Vector v must satisfy one of the following cases.

Case 1: $v \notin \tilde{E}$ and $rank(\tilde{E} + u + v) = rank(E) + 2$. Then $E' = E + \{u, v\}$ satisfies $valid_2(E')$ and $cost_1(E')$ is by 1 smaller than $cost_1(E) = c_{min}$, a contradiction.

Case 2: $v \notin \tilde{E}$ and $rank(\tilde{E} + u + v) = rank(E) + 1$. Because $\tilde{D}$ and $\tilde{E} + u$ are independent sets of vectors, and $\tilde{E} + u + v$ is not, for some $w \in \tilde{E} - \tilde{D}$ the set $\tilde{E} + u + v - w$ is independent. Let e be the set of w in E. One can see that $E' = (E - e + (e - w) + \{u, v\})_2$ satisfies $valid_2(E')$, $cost_1(E') = cost_1(E)$ and $conform(D, E') = conform(D, E) + u + v$, a contradiction.

Case 3: $v \in \tilde{E} - conform(D, E)$. Let e be the set of v in E. One can see that $E' = E - e + (e - v) + \{u, v\}$ gives the same contradiction as in Case 2.

Case 4: $v \in conform(D, E)$, i.e. for some $e \in E$, $v \in d \cap e$ and $\#(d \cap e) \geq 2$. If $e - d$ has at most one element, we obtain a contradiction with $E' = E - e + ((e \cap d) + u)$, as $valid_2(E')$ holds, $cost_1(E') \leq cost_1(E)$ and $conform(D, E') = conform(D, E) + u$. If $e - d$ has at least two elements, we obtain the same contradiction with $E' = E - e + (e - d) + ((e \cap d) + u)$. □

Lemma 4 *Assume $valid_3(D)$ and $rank_2(D) < r_{max}$. Then there exists E such that $valid_3(E)$, $rank_2(E) = rank_2(D) + 1$ and either*
(1) $E = (D - d + (d - w))_3$ for some $w \in d \in D$, or
(2) $E = D + d$ for some triple d of vectors.

Proof. Let F be such that $valid_2(F)$ and $rank(F) = r_{max}$. Among D' such that $valid_2(D')$, $D'_3 = D$ and $rank(D') = rank_2(D)$ choose one that maximizes the size of $conform(D', F)$. Because $rank(F) > rank_2(D) \geq rank(D')$, for some $u \in f \in F$ we have $u \notin span(D')$. Let v be another element of f; this vector must satisfy one of the following cases.

Case 1: $v \notin \tilde{D}'$ and $rank(\tilde{D}' + u + v) = rank(D') + 2$. Then $valid_2(D' + \{u, v\})$, $(D' + \{u, v\})_3 = D$ and $rank(D' + \{u, v\}) = rank(D') + 2$, a contradiction with our choice of D'.

Case 2: $v \notin \tilde{D}'$ and $rank(\tilde{D}' + u + v) = rank(D') + 1$. As in Case 2 of the previous proof, for some $w \in \tilde{D}' - \tilde{F}$, the set $\tilde{D}' + u + v - w$ is independent. Let d be the set of w in D'.

Case 2.1: $\#d = 2$. Then we can replace d in D' with $\{u, v\}$, and this will preserve the property $valid_2(D')$, while the values of D'_3 and $rank(D')$. However the size of $conform(D', F)$ will increase, a contradiction with our choice of D'.

Case 2.2: $\#d > 2$. Then for $D'' = D' - d + (d - w) + \{u, v\}$ we have $valid_2(D'')$ while $rank(D'') = rank(D') + 1 = rank_2(D) + 1$. Therefore we have found $E = D''_3$ such that $rank_2(E) = rank_2(D) + 1$, note that $E = (D - d + (d - w))_3$, which satisfies case (1) of the claim.

Case 3: $v \in \tilde{D}' - conform(D', F)$. We can define $w = v$ and proceed exactly as in case 2.

Case 4: $v \in conform(D', F)$. Let d' be the set of v in D'. If $\#d' > 2$, we can proceed as in Case 2.2, with $d = d'$ and $w = v$. If $\#d' = 2$, then $d' \subseteq f$, consequently for $d = d' + u$ we have $valid_2(D' - d' + d)$ and $rank(D' - d' + d) = rank(D') + 1$. Note that $E = (D' - d' + d)_3$ satisfies case (2) of the claim. □

One consequence of Lemma 4 is that the following algorithm, on input D satisfying $valid_3(D)$, finds E such that $D \subseteq E$, $valid_3(E)$ and $rank_2(E) = rank_3(D)$:

Algorithm 2:

$E \leftarrow D$
while there exists a triple d of vectors such that
    $valid_3(E + d)$ and $rank_2(E + d) > rank_2(E)$ do
  $E \leftarrow E + d$
return E

To see that this algorithm is correct, it suffices to analyze it for $D = \emptyset$. Let us apply Lemma 2, starting from the empty collection of vector sets. Initially, we have $rank_2$ equal to some $2m$. After k iterations consistent with case (2), we obtain $rank_2$ equal to $2m + k$. Suppose that now we can apply case (1) of the Lemma. Then we obtain E consisting of $k - 1$ triples with $rank_2(E) = 2m + k + 1$. This means that a sequence of $k - 1$ triple insertions increases $rank_2$ by $k + 1$, hence one of these insertions increases $rank_2$ by at least 2, which is impossible. Because we $rank_2$ is computable in polynomial time, this algorithm computes $rank_3$ in polynomial time.

Now we have all the tools to analyze the following algorithm for approximating the base cover problem.

Algorithm 3:

(* quick stage *)
$A \leftarrow$ empty collection of vector sets
while there exists d such that $valid(A + d)$ and $\#d \geq 8$
  $A \leftarrow A + d$
(* careful stage *)
for $i \leftarrow 7$ down to 3
  while there exists d such that $valid_i(A + d)$ and $rank_3(A + d) = rank_3(A)$

$A \leftarrow A + d$
(* finishing stage *)
while there exists d such that $valid_2(A + d)$ and $rank_2(A + d) = rank_2(D)$
$A \leftarrow A + d$
while there exists d such that $valid(A + d)$
$A \leftarrow A + d$
return A

Theorem 2 *Algorithm 3 produces in polynomial time a base cover with cost at most* $1 + 1/7 + 1/160$ *times the optimum cost.*

Proof. Say that a collection of vector sets B forms a best solution. We want to find the worst case ratio $cost(A)/cost(B)$. Each time the algorithm selects a set d to the partial solution A, the remaining problem is to find the least cost base cover for $H/span(A)$. In the analysis, we modify B after each selection. Each selection d reduces the dimension of $H/span(A)$ by $\#d$, hence we have to remove $\#d$ vectors from the sets of B. In the same time, the cost of A increases by $1 + \#d$.

During the quick stage the ratio between the increase of $cost(A)$ and the decrease of the remaining $cost(B)$ is at most $9/8$. Later we cannot argue in the same manner, but we have to average the cost increases of the careful stage with the mistake-free finishing stage. We achieve this goal as follows. First, at the beginning of the careful stage we modify B so that it contains the minimum number of singleton sets. By Lemma 3 this can be done without changing the solution cost. Afterwards, during the careful stage, we monitor B_3, while preserving the invariant that $rank_3(B_3)$ is maximal for the linear space $H/span(A)$. During this process, B_3 consists of the "original" sets (with some elements removed) and some "new" triples.

After selecting some d during the careful stage, we remove $\#d$ vectors from the sets of B. If one of these sets becomes empty, then the remaining solution has $\#d = i$ fewer vectors and 1 fewer set, so its costs decreases by $i + 1$, the same as the increase of the cost of A. Otherwise, we need to account for the difference between these two quantities, i.e. 1. We split 1 into i equal parts, and we charge them to the elements of B_3 that had vectors removed.

Now, the size of elements of B_3 drops for two reasons. First, we remove some vectors to maintain the invariant that $\tilde{B}$ is the base of $H/span(A)$. Second, if as a result some sets drop in size to 1, we have to reduce the number of singleton sets in B to the minimum to maintain the second invariant. We do it using the modifications described in Lemma 4. If even one such a modification is of kind (2), it effectively replaces in B a singleton set and a pair with a triple which decreases the cost of B, so there are no charges. A modification of kind (1) removes one vector from a set of B_3, while also replacing a singleton set in B with a pair. As a result, we exchange a size decrease to 1 with a size decrease to at least 2.

Summarizing, a selection of d during the careful stage makes i or fewer decreases of the sets in B_3; if there are fewer than i decreases, this is a step without a charge, otherwise there are exactly i decreases, each associated with a charge of $1/i$. Because we always select the largest possible set, a charge of $1/i$ can be applied only to a set of size at most i, otherwise this set would be selected rather than d. Once a set drops in size to 2, it does not accept further decreases (unless there are no charges).

Now we can see what are possible ratios between the cost of a set in B and the charges it received. Singleton sets and pairs do not accept any charges. A triple can receive one charge, at most $1/3$, and the resulting ratio is at most $(1/3)/4 = 1/12$. A quadruple can receive a most two charges, the first at most $1/4$, when it was reduced in size to a triple, and then at most $1/4$, for the resulting ratio $(1/4 + 1/3)/5 = 7/60$. For the larger sets we get the ratios $(1/5 + 1/4 + 1/3)/6 = 47/360$, $(1/6 + 1/5 + 1/4 + 1/3)/7 = (1 - 1/20)/7 = 1/7 - 1/140$ and $(1/7 + 1/6 + 1/5 + 1/4 + 1/3)/8 = (8/7 - 1/20)/8 = 1/7 - 1/160$. □

4 Given points form a dominating set

Now we consider the case when all edges have length 1 and the set P of the given points forms a dominating set (i.e. each node is in P or is adjacent to P). To assure approximation ratio $4/3$ we may use a version of Algorithm 3. In the quick stage, we first sequentially collapse edges connecting the given points, and then i-stars for $i \geq 4$; one can easily see that if we spend x in the quick stage, then *opt*, the cost of the minimum Steiner tree, is decreased by at least $3x/4$. Therefore we may assume that we deal with a graph G that is an output of the quick stage, and consequently P is an independent set (and a dominating set), and each node in $V - P$ is adjacent to at most 3 nodes from P.

The next 4 lemmas contain the necessary graph-theoretic observations, the fifth lemma provides description on the needed algorithm, (a variation of the careful and finishing stage), and shows the approximation ratio of $4/3$.

For the sake of analysis, let G' be obtained from G by deleting all edges from the graph that are not adjacent to one of the given points. We will say that connected components of G' are V-components, while an intersection of a V-component with P is a P-component.

Lemma 5 *Any minimum spanning tree (MST) of P can be formed as follows: points of each P-component are connected with paths of length 2, and then P components are connected with paths of length 3.*

Proof. No paths of length 1 exist between the given points, because they form an independent set. A path of length 2 between given points means that they share a neighbor, hence that they are in the same P-component. It is easy to see that P-components can be connected by such paths. Afterwards we insert to the tree paths of length 3 as long as possible. Suppose that P is not connected yet, i.e. the paths selected so far form a forest, and any path between two given points from two different trees has at least 4 edges. Let $(v_1, v_2, v_3, \ldots, v_k)$ be a shortest such path. If $v_i \in P$, $1 < i < k$, then either $(v_1, \ldots, v_i)$ or $(v_i, \ldots, v_k)$, joins two trees of the forest, a contradiction because either of the possible both of them are shorter than $(v_1, \ldots, v_k)$. Otherwise $v_3 \notin P$, but some node u is adjacent to v_3, because P is a dominating set. In this case we also get a contradiction, because either (v_1, v_2, v_3, u) or $(v_3, \ldots, v_k)$, joins two trees of the forest. □

We say that a Steiner tree is *normal* if every internal node u is adjacent in the tree to a given point; in proofs we will use $p(u)$ to denote a selected (may be unique) given point adjacent in the tree to u. As before, we *opt* is the minimum cost of a Steiner tree.

Lemma 6 *There exists a normal minimum tree with cost opt.*

Proof. Consider a Steiner minimum tree T with minimum number m of internal nodes not adjacent in the tree to a given point. By the way of contradiction, $m > 0$, i.e. an internal node u is not adjacent in T to a given point. Because P forms a dominating set, G contains an edge $\{u, p\}$ for some $p \in P$. We can insert $\{u, p\}$ to T, and break the resulting cycle by removing an edge adjacent to u to obtain tree T' with the same cost, but with $m - 1$ nodes that are not adjacent in T' to given points, a contradiction. □

We say that a Steiner tree is *special* if (1) it is normal, (2) its full components have at most 4 given points each, (3) its full components have at most one Steiner point, and (4) the union of minimum spanning trees of its full components forms an MST of P. For *any* Steiner tree T we will use a_T and b_T to denote the number of full components with 3 and 4 given points respectively, and exactly one Steiner point. As before, *mst* denotes the cost of a MST of P.

Lemma 7 *The cost of a special Steiner tree T equals $mst - a_T - 2b_T$.*

Proof. Consider a MST S of P that is a union of the minimum spanning trees of full components of an special Steiner tree T. These MSTs have one, two or three edges (paths). An MST with one edge is in $S \cap T$. An MST with two edges (one of a_T many) connects a set of points $p(u), q, r$ where u is the Steiner point of the component; this tree consists of paths $p(u), u, q$ and $p(u), u, r$. It is easy to see that the edge $\{p(u), u\}$ appears twice in S but only once in T. Similarly, a component that connects four given points (one of b_T many) has an edge that appears three times in S. Note that all other edges of these components appear exactly once in both S and T. $\Box$

Lemma 8 *There exists a special Steiner tree T with cost at most $\frac{4}{3}opt - \frac{1}{3}b_T$. tree.*

Proof. Consider a normal Steiner tree T of P with cost *opt* (it exists by Lemma 6). In the fist part of the proof, we will transform it into a Steiner tree satisfying conditions (1), (2) and (3). At an intermediate stage of the transformation let x be the cost of the full components of T that satisfy conditions (2) and (3), y be the cost of the remaining components. At each stage we will decrease y, while satisfying the invariant $x + \frac{4}{3}y + \frac{1}{3}b_T \leq \frac{4}{3}opt$. Observe that this invariant holds initially, because the cost of each component counted by a_T has cost at least 4.

Assume then that $y > 0$ and consider a full component C that violates (2) or (3). Suppose that a Steiner point u in C is adjacent (via edges or paths of length 2 that do not go through a Steiner point) to some four given points, $p(u), q, r$ and s. Because no node is adjacent to more than 3 given nodes, we may assume that the path from u to s has length 2. We may remove the paths from u to q, r and s from C, thus decreasing its cost by some $z \geq 4$, and create a new component consisting of these paths and the edge $\{u, p\}$. In such a stage $\frac{4}{3}y$ decreases by $\frac{4}{3}z \geq z + 1 + \frac{1}{3}$, while x increases by 1 and $\frac{1}{3}b_T$ increases by $\frac{1}{3}$. If we cannot perform a stage like that, we may infer that all full component with exactly one Steiner node satisfy (2) and (3). Suppose then that C contains more than one Steiner point; one of them, say u, is adjacent to exactly one other, say v. We can accomplish a stage by removing from C Steiner point u with the adjacent paths, and creating a new component consisting of these paths and the edge $\{v, p(v)\}$. If the new component contains 3 given points, we decreased y by some $z \geq 3$, and increased x by $z + 1$, and if the new component contains 4 given points, we can repeat the previous calculation.

Now we may assume that $y = 0$ and $x + \frac{1}{3}b_T \leq \frac{4}{3}opt$. Suppose that condition (4) is violated, i.e. the union U of MST's of the full components of T has cost larger than mst (it is clear that U is a spanning tree of P). Let M be an MST of P. By Lemma 5, M consists of paths of length 2 and 3, and because T is normal, U also consists of such paths. Therefore we can insert to U a path of length 2 from M, and from the resulting cycle remove a path of length 3, that belongs to MST of a full component C. If C is just a path, we can replace C with the new path of length 2, decreasing the cost of T. If C contains a Steiner point u, then we removed from U a path of the form $(p(u), u, v)$, and so we can remove path (u, v) from C; because this path has length 2, x remains unchanged, while b_T could only decrease. Therefore we can decrease the cost of U without increasing $x + \frac{1}{3}b_T$. $\Box$

Lemma 9 *Assume that there exists a special Steiner tree T with cost c. Then we can in polynomial time find a special Steiner tree with cost $c + \frac{1}{3}b_T$.*

Sketch of proof. By Lemma 7, $c = mst - a_T - 2b_T$. Therefore it suffices to find a special Steiner tree T' such that $a_{T'} + 2b_{T'} \geq a_T + 2b_T - \frac{1}{3}b_T$.

We will use Algorithm 3 as follows. Our linear space will be over the field $\mathbf{Z_2}$; the base vectors will be nodes and P-components. Vectors in the problem presented to Algorithm 3 will correspond to paths that can be used in MST of P according to Lemma 5. Assume that such a path connects given points p_1 and p_2, which in turn belong to P-components P_1 and P_2. If $P_1 = P_2$ (and the length of this path is 2), we map this path into vector $p_1 + p_2$, otherwise (the case of a path of length 3), we map it into $P_1 + P_2$. Finally, we identify a 3- or 4-star with a set of vectors that correspond to the sets of paths that form MST's of these stars.

It is quite easy to see that Algorithm 3 results in $b'_T \geq \frac{1}{3}b_T$. Moreover, Lemma 4 assures that $2a_{T'} + 3b_{T'} \geq 2a_T + 3b_T$. We obtain the desired result by dividing the above inequalities by 2 and 6 respectively, and adding them together. $\Box$

References

[1] M. Bern and P. Plassman. *The Steiner tree problem with edge lengths 1 and 2.* Inform. Process. Lett. 32: 171–176, 1989.

[2] G. Calinescu, C.G. Fernandes, U.Finkler and H. Karloff. *A better approximation algorithm for finding planar subgraphs.* Proc. 7th SODA: 16–25, 1996.

[3] H.N. Gabow and M. Stallmann. *Efficient algorithms for graphic matroid intersection and parity.* Proc. ICALP'85. Lecture Notes in Computer Science, 194: 210–220, 1985.

[4] L. Lovász and M.D. Plummer. *Matching Theory.* Elsevier Science, Amsterdam, 1986.

[5] Orlin and Vande Vate, *Solving the Linear Matroid Parity Problem as a Sequence of Matroid Intersection Problems.* MATHPROG: Mathematical Programming 47, 1990.

[6] V. J. Rayward-Smith, *The computation of nearly minimal Steiner trees in graphs.* International J. Math. Ed. Sci. Tech. 14: 15–23, 1983.