Finding Well-Balanced Pairs of Edge-Disjoint Trees in Edge-Weighted Graphs

Jørgen Bang-Jensen [a,*] Daniel Goncalves [b] Inge Li Gørtz [c]

[a] *Department of Mathematics and Computer Science, University of Southern Denmark, DK-5230 Odense, Denmark*

[b] *LabRI, Université Bordeaux 1, 33405 Talence, France*

[c] *Department of Informatics and Mathematical Modeling, The Technical University of Denmark, DK 2800 Lyngby, Denmark*

Abstract

The well known number partition problem is NP-hard even in the following version: Given a set S of n non-negative integers; partition S into two sets X and Y such that $|X| = |Y|$ and the sum of the elements in X is as close as possible to the sum of the elements in Y (or equivalently, minimize the maximum of the two sums). In this paper we study the following generalization of the partition problem: given an edge-weighted graph G containing two edge-disjoint spanning trees. Find a pair of edge-disjoint spanning trees such that the maximum weight of these two trees is as small as possible. In the case when G is precisely the union of two trees this problem may be seen as a generalization of the partition problem in which we have added a graph structure to the numbers (through the edges) and the extra restriction that only sets X and Y which correspond to trees in G are valid partitions. We first show how to obtain a 2-approximation via an algorithm for weighted matroid partition. Then we describe a simple heuristic which when applied to the 2-approximation above will result in a solution whose value is no more than $\frac{3}{2}$ times the value of an optimal solution. We also show that the approach above may sometimes exclude all optimal solutions. Both the partition problem and its generalization to the problem above on edge-disjoint spanning trees are special cases of the problem of finding, in a weighted matroid with two disjoint bases, a pair of disjoint bases which minimize the maximum of their weights. In the last part of the paper we give some results on this problem for transversal matroids which turn out to be analogous to those for graphs.

Key words: Edge-disjoint spanning trees, approximation algorithm, graphic matroid, transversal matroid, matroid partition algorithm, approximation scheme.

* Corresponding author.
Email addresses: jbj@imada.sdu.dk (Jørgen Bang-Jensen), goncalve@labri.fr

1 Introduction

In communication networks that are to be shared by several parties, a desirable property would be that these parties could use disjoint parts of the network, hence avoiding interference of messages and retaining privacy. One example of such a situation could be when two phone companies share the same optical network. Here it would be useful to be able to divide the cables of the network into two parts A, B plus maybe some unused cables C such that company 1 used part A, company 2 part B and C is left for other purposes. Let us assume that both companies wish to be able to service all possible clients (meaning that A and B must be connected networks and span all clients) and that no company wishes to pay more than absolutely necessary for their network (in particular, it does not want to pay much more than the other company). Here the price to pay is a fixed price for each section of the optical network and varies according to the length and other properties of the particular section. The problem above can be formulated as the following graph theoretical problem. Note that if a graph contains two edge-disjoint spanning subgraphs, then in particular it contains two edge-disjoint trees and we make the reasonable assumption that no cable price is negative.

Problem 1.1 *Given a graph G containing two edge-disjoint spanning trees and non-negative real valued weight function ω on the edges. Find a pair of edge-disjoint trees $T, \tilde{T}$ which minimizes*[1] $\max\{\omega(T), \omega(\tilde{T})\}$.

In the next section we show, by a straightforward reduction from the partition problem, that Problem 1.1 is NP-hard, even in the case when G is just the union of two spanning paths. In fact one may view Problem 1.1 as a restricted partition problem in which, besides the numbers, an underlying graph structure is given and valid partitions must correspond to spanning trees in the graph. One can easily generalize the partition problem to the following NP-hard problem which we call **Partition($2k$)**: Given a set S of $n \geq 2k$ non-negative integers. Find two disjoint sets $X, Y \subset S$ each of size k so that $\max\{\sum_{s\in X} s, \sum_{t\in Y} t\}$ is as small as possible. It is not difficult to see that this problem contains the partition problem as a special case (see the proof of Proposition 2.1 below). How does one solve this problem heuristically? In particular, how do we find the right set of $2k$ elements to partition? The answer is easy: first sort the elements of S according to increasing value and consider the first $2k$ elements only. Now apply your best heuristic for the partition problem (into sets of equal size) to this part. It is not difficult to show that this reduction cannot remove every optimal solution to the original problem (for a proof see Section 7) and hence we may safely (and quickly!) reduce to

(Daniel Goncalves), ilg@imm.dtu.dk (Inge Li Gørtz).

[1] As usual $\omega(T)$ is the sum of ω over all edges in T

this potentially much smaller problem.

Inspired by this, let us go back to the problem of the phone companies and see what a similar idea would be here. If instead the phone companies collaborated and wanted to get disjoint connected networks so that the total price of the two networks was minimized, we would have the following graph problem.

Problem 1.2 *Given a graph G containing two edge-disjoint spanning trees and with edge weights $\omega : E \to R_+$. Find a pair of edge-disjoint spanning trees T_1, T_2 which minimizes $\omega(T_1) + \omega(T_2)$.*

We show in the next section that Problem 1.2 is polynomially solvable. Since every solution to Problem 1.1 consists of two edge-disjoint spanning trees, it is easy to see that the following lower bound is valid. Here $\mathrm{OPT}_{\mathrm{minmax}}$ and $\mathrm{OPT}_{\mathrm{minsum}}$ denote the value of the optimal solution to Problems 1.1 and 1.2, respectively.

$$\mathrm{OPT}_{\mathrm{minmax}} \geq \left\lceil \frac{\mathrm{OPT}_{\mathrm{minsum}}}{2} \right\rceil . \tag{1}$$

Below we will use the phrase "the **min sum reduction**" for the approach where we replace the edges of the starting graph by the edges of two edge-disjoint spanning trees which form an optimal solution to Problem 1.2. By the observation above we have

Proposition 1.1 *Let G be a graph containing two edge-disjoint spanning trees and let ω be a non-negative weight function on the edges of G. Applying the min sum reduction and taking any pair of edge-disjoint trees in the resulting graph gives a solution to Problem 1.1 whose value is at most twice the optimum.*

In this paper we show how one can obtain a $\frac{3}{2}$-approximation algorithm for Problem 1.1 by first performing the min sum reduction and then applying a heuristic for Problem 1.1 on the graphs which are the union of two spanning trees. We also give an example which shows that the approach of working only on the edges of an optimum solution to Problem 1.2 may result in an optimality gap of $\frac{7}{6}$. On the other hand, in [1] we show that for a large variety of instances the min sum reduction not only preserves the value of an optimal solution but also make Problem 1.1 much easier to solve by (meta-)heuristics. Finally, we consider the further generalization of the partition problem to matroids. As we will see below Problem 1.1 can be formulated as a special case of a more general problem of finding disjoint bases in a matroid such that the maximum weight of the two bases is minimized. Also for this problem the min sum reduction is valid and corresponds to finding two disjoint bases whose total weight is as small as possible. Thus also for general matroids we obtain a 2-approximation to the min max problem via the min sum solution. Another

class of matroids of practical interest is the class of transversal matroids. In the last part of the paper we give some results on the analogue of Problem 1.1 for transversal matroids.

2 Terminology and Preliminaries

In this paper all graphs may have parallel edges but no loops. A loop less graph is **simple** if it has no parallel edges. We use standard terminology on graphs such as that used in [2,12].

2.1 Matroids

We start by recalling the definition of a matroid. For terminology on matroids not defined below we refer to [10,11]. A **matroid** $\mathcal{M}$ consists of a set S (called the ground set) and a collection $\mathcal{F}$ of subsets of S, satisfying the following two axioms:

(A1) If $Y \in \mathcal{F}$ then $\forall X \subseteq Y$: $X \in \mathcal{F}$.
(A2) $X, Y \in \mathcal{F}$ and $|X| > |Y| \Rightarrow \exists x \in X - Y$: $Y + x \in \mathcal{F}$ [2].

The subsets in $\mathcal{F}$ are called **independent** and all other subsets of S are **dependent**. A **circuit** is a minimal dependent set. A **base** is a maximal [3] independent set. It is an easy consequence of (A2) that all bases of $\mathcal{M}$ have the same cardinality.

When we speak of algorithms for a matroid $\mathcal{M}$ it often makes no sense to assume that we have a list of the independent sets of $\mathcal{M}$. Instead we assume that we have at our disposal an **independence oracle** $\mathcal{O}$ which given a subset X of the ground set will return the answer "yes" if and only if X is independent in $\mathcal{M}$. If X is dependent we also assume that $\mathcal{O}$ will return a circuit $C \subseteq X$. A matroid algorithm $\mathcal{A}$ is polynomial if it makes only a polynomial number of calls (measured in the size of the ground set S) to an independence oracle and all other operations of $\mathcal{A}$ can be bounded by a polynomial in $|S|$. An independence oracle is **polynomial** if it can provide its answer in polynomial time with respect to the size of ground set.

Suppose that besides the matroid $\mathcal{M} = (S, \mathcal{F})$ we also have a weight function ω on S. By an **optimal base** of $\mathcal{M}$ (w.r.t. ω) we mean a base B such that $\omega(B)$ is minimum over all bases of $\mathcal{M}$. Among many nice properties of matroids the

[2] Here and in the rest of the paper we use the shorthand notation $Y + x$ for $Y \cup \{x\}$.
[3] both minimal and maximal are with respect to inclusion

following is the most useful: one can find an optimal base of $\mathcal{M}$ by sorting the elements of S according to increasing weight $\omega(s_1) \leq \omega(s_2) \leq \ldots \ldots \omega(s_{|S|})$ and then starting from $B = \emptyset$, iterate through the elements in sorted order and adding s_i to the current B if and only if $B + s_i$ is independent. This algorithm, called the **greedy** algorithm for matroids, always returns an optimal base (see e.g. [10, Section 7.4]).

Theorem 2.1 (Matroid union [11]) *Let $\mathcal{M}_1 = (S, \mathcal{F}_1), \mathcal{M}_2 = (S, \mathcal{F}_2)$ be matroids on the same ground set S. Define $\mathcal{F}$ to be the following collection of subsets of S: $X \in \mathcal{F}$ if and only if there exists a partition $X = X_1 \cup X_2$ such that $X_i \in \mathcal{F}_i$ for $i = 1, 2$. Then $\mathcal{M} = (S, \mathcal{F})$ is a matroid called the* **union** *of $\mathcal{M}_1$ and $\mathcal{M}_2$ and is denoted by $\mathcal{M} = \mathcal{M}_1 \vee \mathcal{M}_2$.*

Theorem 2.2 *[10, Theorem 13.1.1] Given matroids $\mathcal{M}_1 = (S, \mathcal{F}_1), \mathcal{M}_2 = (S, \mathcal{F}_2)$ and polynomial independence oracles for $\mathcal{M}_1, \mathcal{M}_2$ we can check in polynomial time whether a given set $X \subset S$ is independent in $\mathcal{M}_1 \vee \mathcal{M}_2$.*

Combining this with the fact that the greedy algorithm is polynomial when we have a polynomial independence oracle we get the following.

Theorem 2.3 *[10, Chapter 13] There is a polynomial algorithm for checking whether a matroid $\mathcal{M} = (S, \mathcal{F})$ with a polynomial independence oracle $\mathcal{O}$ has two disjoint bases and in case the answer is "yes" and ω is an arbitrary weight function on S we can find in polynomial time a pair of disjoint bases B_1, B_2 minimizing $\omega(B_1) + \omega(B_2)$ over all pairs of disjoint bases.*

2.2 Edge-Disjoint Spanning Trees in Graphs

Now we are ready to show how Problem 1.2 can be solved via matroid algorithms. Given a graph $G = (V, E)$ we can define a matroid $\mathcal{M}(G)$ (called the **circuit matroid** of G) by taking E as the ground set and as independent sets those subsets of E which induce an acyclic graph. It is easy to see that $\mathcal{M}(G)$ is indeed a matroid and that the bases of $\mathcal{M}(G)$ are precisely the maximal spanning forests of G (which are the spanning trees provided G is connected). Thus the following is a direct consequence of Theorem 2.3.

Theorem 2.4 *Problem 1.2 is polynomially solvable.*

The following lemmas (which also hold generally for bases of matroids) are easy to prove (see e.g. [12, Exercises 8.2.17 and 8.2.21]

Lemma 2.1 *Let T and T' be edge-disjoint spanning trees and let $e \in E(T), e' \in E(T')$. Then $T + e' - e$ and $T' + e - e'$ are (edge-disjoint) spanning trees if and*

only if e' is on the fundamental cycle[4] *of $T' + e$ and e is on the fundamental cycle of $T + e'$.*

Lemma 2.2 *Let T_1 and T_2 be edge-disjoint spanning trees. Then for every $e_i \in T_i$, $i = 1, 2$, there exists at least one edge $e_{3-i} \in T_{3-i}$ such that $T_i - e_i + e_{3-i}$ and $T_{3-i} - e_{3-i} + e_i$ are (edge-disjoint) spanning trees (possibly $e_i = e_{3-i}$).*

Definition 2.1 *A 2T-graph is the union of two trees having the same vertex set. In particular every 2T-graph on n vertices has $2n - 2$ edges and may contain parallel edges.*

We observe that since every vertex in a 2T-graph G has at least one edge in each tree, we have $\delta(G) \geq 2$.

Proposition 2.1 *Problem 1.1 is NP-hard.*

PROOF. The partition problem is NP-hard even in the following version (see [6]): Given a set S of $2n$ integers $S = \{x_1, y_1, x_2, y_2, \ldots, x_n, y_n\}$ find a partition of S into two sets S_1, S_2 of n integers each so that no S_i contains both x_j and y_j for some j and $|\sum_{x \in S_1} x - \sum_{x \in S_2} x|$ is minimized[5]. Given such an instance we construct an instance of Problem 1.1 by letting G be the graph with $n + 1$ vertices $\{v_0, v_1, \ldots, v_n\}$ and two edges between v_{i-1} and v_i with weights x_i and y_i respectively for each $i = 1, 2, \ldots, n$. Clearly any solution $T, \bar{T}$ to Problem 1.1 corresponds to a valid partitioning of S and vice versa and optimal solutions are preserved by the transformation. □

Note that the proof of Proposition 2.1 shows that Problem 1.1 is NP-hard even when G is the union of two spanning paths and hence also for 2T-graphs.

3 A heuristic for partitioning the edges of a 2T-Graph evenly

In this section the weight of an edge may be any real number.

Definition 3.1 *A d-vertex is a vertex of degree d.*

First observe that every 2T-graph has a 2-vertex or a 3-vertex. This follows from the facts that there is no vertex of degree less than two and the sum of

[4] If T is a tree and uv is an edge not in T joining two vertices u, v of T, then the **fundamental cycle** of $T + uv$ is the unique cycle formed by the uv-path in T and the edge uv.

[5] Given a normal instance $\{s_1, s_2, \ldots, s_n\}$ of the partition problem simply let $x_i = s_i$ and $y_i = 0$.

the degrees is $4n-4$.

Theorem 3.1 *The edge set of every 2T-graph G with weights ω on the edges can be partitioned into two spanning trees T_h and T_l, with $\omega(T_h) \geq \omega(T_l)$, such that*

$$\omega(T_h) - \omega(T_l) \leq \max_{e \in T_h} \omega(e) - \min_{e \in T_l} \omega(e) \ . \tag{2}$$

PROOF. The theorem is true if G is of order 2. We now prove the claim by induction on the order of G. Assume that we have proved the claim for 2T-graphs of order $k < n$. Let H be a 2T-graph on n vertices and let T_1 and T_2 be two spanning trees partitioning the edges of H.

Case 1: there is a 2-vertex v in H. Let e_1 and e_2 be the two edges adjacent to v such that $\omega(e_1) \geq \omega(e_2)$ and consider the graph $H' = H - v$. Since $\deg_{T_1}(v) = 1$ and $\deg_{T_2}(v) = 1$ the graphs $T_1 - v$ and $T_2 - v$ are connected spanning trees in H'. So the graph H' is a 2T-graph with order smaller than H. By induction, there are two trees T'_h and T'_l partitioning H' with $\omega(T'_h) \geq \omega(T'_l)$ such that $\Omega' = \omega(T'_h) - \omega(T'_l) \leq \max_{e \in T'_h} \omega(e) - \min_{e \in T'_l} \omega(e)$.

Now let $T_1 = T'_l + e_1$ and $T_2 = T'_h + e_2$. We have that $|\omega(T_2) - \omega(T_1)| = |\omega(T'_h) + \omega(e_2) - \omega(T'_l) - \omega(e_1)|$, and so, $|\omega(T_2) - \omega(T_1)| = |\Omega' - (\omega(e_1) - \omega(e_2))|$. We finally obtain that $-(\omega(e_1) - \omega(e_2)) \leq \omega(T_2) - \omega(T_1) \leq \Omega'$. Thus, if $\omega(T_1) \geq \omega(T_2)$ then

$$\omega(T_1) - \omega(T_2) \leq \omega(e_1) - \omega(e_2) \leq \max_{e \in T_1} \omega(e) - \min_{e \in T_2} \omega(e) \ ,$$

since $e_1 \in T_1$ and $e_2 \in T_2$. Similarly, if $\omega(T_2) \geq \omega(T_1)$ then

$$\omega(T_2) - \omega(T_1) \leq \Omega' \leq \max_{e \in T'_h} \omega(e) - \min_{e \in T'_l} \omega(e) \leq \max_{e \in T_2} \omega(e) - \min_{e \in T_1} \omega(e) \ ,$$

where we used that $\max_{e \in T_2} \omega(e) = \max\{\max_{e \in T'_h} \omega(e), \omega(e_2)\}$ and $\min_{e \in T_1} \omega(e) = \min\{\min_{e \in T'_l} \omega(e), \omega(e_1)\}$. Thus the partition of H into T_1 and T_2 satisfies (2) (with the naming chosen so that $T_h = T_1$ if $\omega(T_1) \geq \omega(T_2)$ and $T_h = T_2$ otherwise).

Case 2: there is a 3-vertex v in H. Without loss of generality v has degree one in T_1. Let $e_1 = vv_1$ be the edge incident to v in T_1 and let $e_2 = vv_2$ and $e_3 = vv_3$ be the edges incident to v in T_2. By Lemma 2.2 we may assume that

$$\omega(e_1) \leq \max\{\omega(e_2), \omega(e_3)\} \ . \tag{3}$$

Consider the graph $H' = H - v + v_2v_3$ obtained by deleting v and adding a new edge v_2v_3 and setting $\omega(v_2v_3) = \omega(e_2) + \omega(e_3) - \omega(e_1)$. Since the graphs $T_1 - v$ and $T_2 - v + v_2v_3$ are edge-disjoint trees and cover H', the graph H' is a 2T-graph with order smaller than H. By induction, there are two trees T'_h and T'_l partitioning H' with $\omega(T'_h) \geq \omega(T'_l)$ and such that $\omega(T'_h) - \omega(T'_l) \leq \max_{e\in T'_h} \omega(e) - \min_{e\in T'_l} \omega(e)$.

Case 2A: the edge v_2v_3 is in T'_h. Let $T_l = T'_l + e_1$ and $T_h = T'_h + e_2 + e_3 - v_2v_3$. Then, $\omega(T_l) = \omega(T'_l) + \omega(e_1)$ and $\omega(T_h) = \omega(T'_h) + \omega(e_1)$. Let $\Omega = \omega(T_h) - \omega(T_l)$. It is clear that $\Omega = \omega(T'_h) - \omega(T'_l) \leq \max_{e\in T'_h} \omega(e) - \min_{e\in T'_l} \omega(e)$. If $\max_{e\in T_h} \omega(e) \geq \max_{e\in T'_h} \omega(e)$, then (2) holds for T_h and T_l, so we assume that $\omega(v_2v_3) > \max_{e\in T_h} \omega(e)$. Since $\omega(v_2v_3) = \omega(e_3) + \omega(e_2) - \omega(e_1)$ and we have $\max_{e\in T_h} \omega(e) \geq \max\{\omega(e_2), \omega(e_3)\}$ we know that $\min\{\omega(e_2), \omega(e_3)\} > \omega(e_1)$. By Lemma 2.1, we can exchange the edge e_1 with one of e_2 or e_3 to get two new edge-disjoint spanning trees. Without loss of generality assume that we may exchange e_1 and e_2. Consider the trees $T^*_l = T_l + e_2 - e_1$ and $T^*_h = T_h + e_1 - e_2$.

Now it is clear that

$$\omega(T^*_h) - \omega(T^*_l) = \Omega + 2\omega(e_1) - 2\omega(e_2) \; . \tag{4}$$

Assume first that $\omega(T^*_h) > \omega(T^*_l)$ holds. Then we have

$$\omega(T^*_h) - \omega(T^*_l) \leq \max_{e\in T'_h} \omega(e) - \min_{e\in T'_l} \omega(e) + 2\omega(e_1) - 2\omega(e_2) \; . \tag{5}$$

Since $\max_{e\in T'_h} \omega(e) = \omega(v_2v_3)$ and $\omega(e_1) < \omega(e_2)$ we have

$$\begin{aligned} \omega(T^*_h) - \omega(T^*_l) &\leq \omega(e_2) + \omega(e_3) - \omega(e_1) - \min_{e\in T'_l} \omega(e) + \omega(e_1) - \omega(e_2) && (6)\\ &= \omega(e_3) - \min_{e\in T'_l} \omega(e) && (7)\\ &\leq \max_{e\in T^*_h} \omega(e) - \min_{e\in T^*_l} \omega(e) \; . && (8) \end{aligned}$$

Hence (2) holds for the pair (T^*_h, T^*_l).

Suppose now that $\omega(T^*_l) > \omega(T^*_h)$. Define the number M by

$$M = \sum_{e\in E(T_h)-e_2} \omega(e) - \sum_{e\in E(T_l)-e_1} \omega(e) \; . \tag{9}$$

Then

$$\omega(T_h) - \omega(T_l) = M + \omega(e_2) - \omega(e_1) \ . \tag{10}$$

$$\omega(T_l^*) - \omega(T_h^*) = -M + \omega(e_2) - \omega(e_1) \ . \tag{11}$$

Now it follows easily that (2) holds for one of the pairs (T_h, T_l) or (T_l^*, T_h^*). Recall that $\omega(T_h) - \omega(T_l) \geq 0$. Hence if $M \leq 0$ (2) holds for (T_h, T_l) and otherwise it holds for (T_l^*, T_h^*)).

Case 2B: the edge v_2v_3 is in T_l'. Let $T_h = T_h' + e_1$ and $T_l = T_l' + e_2 + e_3 - v_2v_3$. Let $\Omega = \omega(T_h) - \omega(T_l)$ and observe as above that $\Omega = \omega(T_h') - \omega(T_l') \leq \max_{e \in T_h'} \omega(e) - \min_{e \in T_l'} \omega(e)$. By (3) $\omega(v_2v_3) = \omega(e_2) + \omega(e_3) - \omega(e_1) \geq \min\{\omega(e_2), \omega(e_3)\}$. Hence $\min_{e \in T_l} \omega(e) \leq \min_{e \in T_l'} \omega(e)$ and (2) holds for T_h and T_l. □

4 A Better Approximation Algorithm for Problem 1.1

Lemma 4.1 *Let G be a 2T-graph and let T_h, T_l be a partitioning of G into 2-edge disjoint spanning trees such that (2) holds. Then*

$$\omega(T_h) \leq \frac{3}{2} \max \left\{ \max_{e \in G} \omega(e), \frac{\omega(G)}{2} \right\} . \tag{12}$$

PROOF. By (2) and the fact that $\omega(T_h) + \omega(T_l) = \omega(G)$ we have

$$\begin{aligned} 2\omega(T_h) &= (\omega(T_h) + \omega(T_l)) + (\omega(T_h) - \omega(T_l)) && (13)\\ &\leq \omega(G) + (max_{e \in T_h} \omega(e) - \min_{e \in T_l} \omega(e)) && (14)\\ &\leq \omega(G) + \max_{e \in G} \omega(e) \ . && (15) \end{aligned}$$

Let e' be chosen such that $\omega(e') = \max_{e \in G} \omega(e)$. If $\omega(e') \leq \frac{\omega(G)}{2}$ then (15) implies that $2\omega(T_h) \leq \frac{3}{2}\omega(G)$ and thus (12) holds. Now assume that $\omega(e') > \frac{\omega(G)}{2}$. Then (15) implies that $2\omega(T_h) \leq 3\omega(e')$ showing that (12) holds again. □

Lemma 4.2 *Let G be an edge-weighted graph with weight function ω, such that G contains two edge-disjoint spanning trees and let T_1, T_2 be a solution to Problem 1.2 on G. Then for every pair (T, T') of edge-disjoint trees in G we have*

$$\max_{e \in T \cup T'} \omega(e) \geq \max_{e \in T_1 \cup T_2} \omega(e) \ . \tag{16}$$

PROOF. This follows from the fact that $(T_1 \cup T_2)$ is an optimal base in the matroid $\mathcal{M} = \mathcal{M}(G) \vee \mathcal{M}(G)$ (see e.g. [10, page 153]). □

Theorem 4.1 *There exists a $\frac{3}{2}$-approximation algorithm for Problem 1.1.*

PROOF. As we argued in the introduction, given any edge-weighted graph H with two edge-disjoint spanning trees, we can find an optimal solution T, T' to Problem 1.2 in polynomial time. Now let G be the spanning 2T-graph of H consisting of precisely the edges of T and T'. Clearly, for every solution (T_1, T_2) to Problem 1.1 on H we have $\omega(T_1) + \omega(T_2) \geq \omega(T) + \omega(T') = \omega(G)$, so

$$\max\{\omega(T_1), \omega(T_2)\} \geq \frac{\omega(G)}{2} \,. \tag{17}$$

Let (T_1^*, T_2^*) be an optimal solution to Problem 1.1 on the graph H. By Lemma 4.2 and the fact that (T_1^*, T_2^*) satisfies (17) we have

$$\max\{\omega(T_1^*), \omega(T_2^*)\} \geq \max\left\{ \max_{e \in G} \omega(e), \frac{\omega(G)}{2} \right\} . \tag{18}$$

Now applying Lemma 4.1 we get that for any pair (T_h, T_l) of edge-disjoint trees in G satisfying (2) we have

$$\begin{aligned} \omega(T_h) &\leq \frac{3}{2} \max\left\{ \max_{e \in G} \omega(e), \frac{\omega(G)}{2} \right\} && (19) \\ &\leq \frac{3}{2} \max\{\omega(T_1^*), \omega(T_2^*)\} \,. && (20) \end{aligned}$$

It remains to show that in G we can find a pair T_h, T_l satisfying (2) in polynomial time. It is easy to see that the proof of Theorem 3.1 can be turned into a polynomial algorithm **split** for finding two trees T_h and T_l which satisfy (2). The algorithm is described in Algorithm 1 below. The main observation is that by doing $O(n)$ work at each step we can obtain that (2) holds in each step as we construct the two trees bottom up. The complexity of the algorithm is clearly at most $O(n^2)$. Hence the total complexity of the approximation algorithm is dominated[6] by that of finding an optimal solution to Problem 1.2.

□

[6] The algorithm for finding an optimal solution to Problem 1.2 makes $O(n)$ calls to an independence oracle which is implemented as a search for a certain path in a graph of size at least $O(n)$. See [10, Chapter 13] for details

Algorithm 1. : Split
Input: A $2T$-graph H and two edge-disjoint spanning trees T_1, T_2 of H

for $j = 1$ to $n - 1$ **do**
  if H has a 2-vertex u **then**
    Remove u from H and update T_1 and T_2 by removing the edge incident to u from both.
  else $\{H$ has a 3-vertex $u\}$
    Choose i such that u is a 1-vertex in T_i and remove the edge zu incident to u from T_i.
    Remove the edges xu and yu incident with u in T_{3i-1}
    Add a new edge xy with weight $\omega(xu) + \omega(yu) - \omega(zu)$ to T_{3i-1}.
  end if
end for
Set $T_h = T_i$ and $T_l = T_{3-i}$ where $\omega(T_i) = \max\{\omega(T_1), \omega(T_2)\}$
Denote by v_j the vertex removed from H in the jth iteration above.
for $j = n - 1$ to 1 **do**
  if v_j had degree 2 when removed from H **then**
    Add the heaviest of the two removed edges to T_l and the other to T_h.
  else $\{v_j$ had degree 3 when removed from $H\}$
    Let e_1, e_2, and e_3 be the removed edges and let e be the edge that replaced e_2 and e_3.
    Replace e by e_2 and e_3 in the tree containing e and add e_1 to the other tree.
    If it is possible to obtain a lower value of the heaviest tree by exchanging e_1 with either e_2 and e_3 (while ensuring we have two spanning trees) then do this.
  end if
  Rename T_h and T_l such that T_h is the heaviest of the two trees.
end for

5 How Good is the Min Sum Reduction?

As we saw above we can get a $\frac{3}{2}$-approximation algorithm for Problem 1.1 by first applying the min sum reduction and then applying the algorithm **split**.

The example in Figure 1 shows that in some cases the min sum reduction may remove all optimal solutions to Problem 1.1. Let $T_1 = \{v_1v_5, v_2v_4, v_3v_4, v_4v_5\}$, $T_2 = \{v_1v_2, v_1v_3, v_1v_4, v_2v_5\}$, $T_h = \{v_1v_3, v_2v_4, v_3v_4, v_3v_5\}$ and finally $T_l = \{v_1v_4, v_2v_3, v_2v_5, v_4v_5\}$. Then (T_1, T_2) and (T_h, T_l) are both pairs of edge-disjoint spanning trees. The weights of these trees are $\omega(T_1) = 2k + p + 1$, $\omega(T_2) = k + 2p + 1$, $\omega(T_h) = 2k + 2$, $\omega(T_l) = k + 3p + \alpha$.

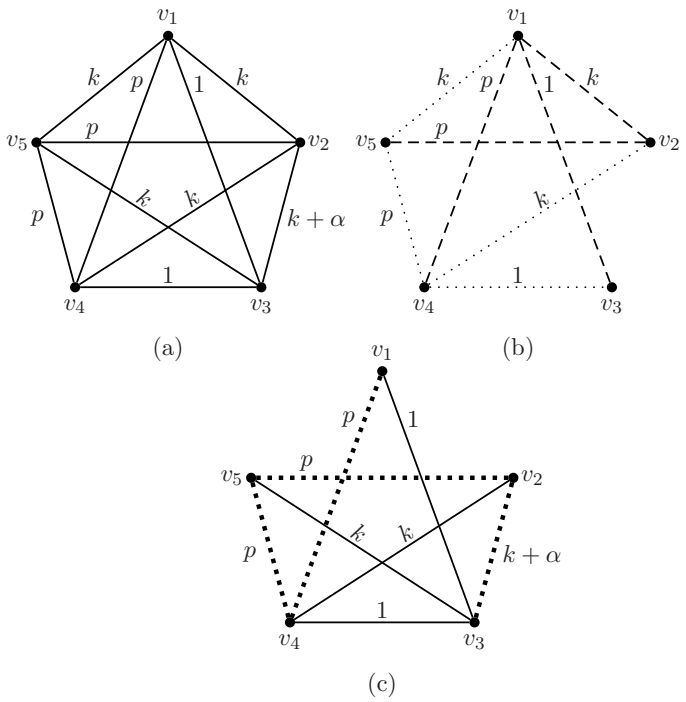

Fig. 1. In (a) the whole graph is shown. The weights should satisfy that $k > p \geq 1$. In (b) the optimum solution to Problem 1.2 is shown. The dotted and dashed edges represent the trees T_1 and T_2, respectively. In (c) the optimum solution to Problem 1.1 is shown. The solid and fat dotted edges represent the trees T_h and T_l, respectively.

Note that (T_1, T_2) is an optimal solution to Problem 1.2 since there are only 5 edges with a cost lower than k and these are all included in $T_1 \cup T_2$. It is also easy to verify that (T_1, T_2) is an optimal solution to Problem 1.1 on the 2T-graph $T_1 \cup T_2$. We will now show that by choosing the weights appropriately we can exclude all optimal solutions to Problem 1.1 by performing the min sum reduction to reduce the instance to the optimal min sum solution $T_1 \cup T_2$.

We want to ensure $\omega(T_2) \leq \omega(T_1)$, $\omega(T_l) \leq \omega(T_h)$, $\omega(T_h) < \omega(T_1)$ and $\omega(T_1) + \omega(T_2) < \omega(T_h) + \omega(T_l)$. If we set $\alpha = 1$, we need $1 < p \leq k/3$. If we set $p = k/3$ we get:

$$\omega(T_1) = \frac{7}{3}k + 1 \quad \text{and} \quad \omega(T_h) = 2k + 2 \ ,$$

which gives us a lower bound of

$$\frac{\omega(T_1)}{\omega(T_h)} = \frac{\frac{7}{3}k + 1}{2k + 2} \to \frac{7}{6} \text{ as } k \to \infty \ .$$

Above we had $\omega(T_h) + \omega(T_l) - (\omega(T_1) + \omega(T_2)) = 4k + 3 - (4k + 2) = 1$. If instead we set $\alpha = k - 4$ and $p = 2$ we get a smaller difference between $\omega(T_1) = 2k + 3$ and $\omega(T_h) = 2k + 2$, but a bigger difference between the sums:

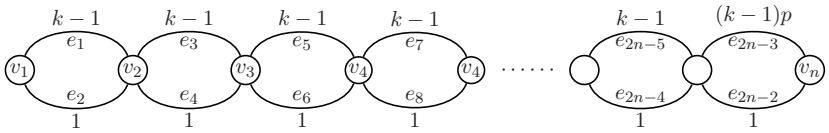

Fig. 2. Here k is an integer greater than 2 and p is the number of nodes minus two (p should be even). The optimum solution is to set $T_1 = \{e_1, e_3, e_5, \ldots, e_{2n-5}, e_{2n-2}\}$ and $T_2 = \{e_2, e_4, e_6, \ldots, e_{2n-4}, e_{2n-3}\}$. In worst case the algorithm **split** gets the solution $T_1 = \{e_1, e_4, e_5, e_8, \ldots, e_{2n-3}\}$ and $T_2 = \{e_2, e_3, e_6, e_7, \ldots, e_{2n-2}\}$.

$\omega(T_h) + \omega(T_l) - (\omega(T_1) + \omega(T_2)) = 4k + 4 - (3k + 8) = k - 4$. Thus the proportion between the total weight of a min sum solution and an optimal min max solution may be almost $\frac{4}{3}$.

6 The Case of 2T-Graphs

For 2T-graphs there is no prepossessing step (via a min sum solution) before we apply algorithm **split**, so the question is whether that will lead to a better approximation guarantee for the algorithm **split.**

If we just take the generic version of **split** then we can show a lower bound of 3/2 for the algorithm. For the graph in Figure 2 the optimum solution is to set $T_1 = \{e_1, e_3, e_5, \ldots, e_{2n-5}, e_{2n-2}\}$ and $T_2 = \{e_2, e_4, e_6, \ldots, e_{2n-4}, e_{2n-3}\}$. This gives $w(T_1) = p(k-1) + 1 = pk - p + 1$ and $w(T_2) = p + (k-1)p = pk$.

Since we have not given any order for which the algorithm picks the vertices to of degree 2 (there are none of degree 3) to remove, we can assume the worst case scenario. Assume the algorithm removes the vertices from the right, i.e., first v_n, then v_{n-1} and so on down to v_1. In the first rebuilding step we set $T_h = \{e_1\}$ and $T_l = \{e_2\}$. In the next step we balance best possible and get $T_h = \{e_1, e_4\}$ and $T_l = \{e_2, e_3\}$, in the third step $T_h = \{e_1, e_4, e_5\}$ and $T_l = \{e_2, e_3, e_6\}$, and so on. Before the final step we have $w(T_h) = w(T_l) = (p/2)k$. Thus, after the final step we have $T_h = \{e_1, e_4, e_5, e_8, \ldots, e_{2n-3}\}$ and $T_l = \{e_2, e_3, e_6, e_7, \ldots, e_{2n-2}\}$. This gives $w(T_h) = (p/2)k + (k-1)p = (p/2)(3k-2)$ and $w(T_l) = (p/2)k + 1$. This gives a lower bound of

$$\frac{(p/2)(3k-2)}{pk} = \frac{3k-2}{2k} = 3/2 - 1/k\ .$$

To get a better approximation guarantee than 3/2 we will try to give a strategy for picking the nodes of degree 2 or 3. We will call this strategy **keep-max** Instead of taking first the nodes of degree 2 and then 3 in arbitrary order when

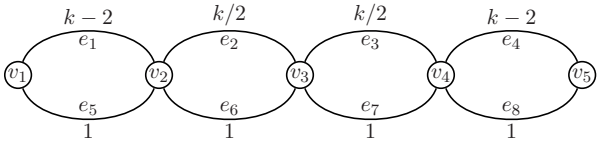

Fig. 3. Here k is an integer greater than 3. The optimum solution is to set $T_1 = \{e_1, e_2, e_7, e_8\}$ and $T_2 = \{e_3, e_4, e_5, e_6\}$. This gives $w(T_1) = w(T_2) = 3k/2$. The algorithm gets $T_1 = \{e_1, e_6, e_7, e_4\}$ and $T_2 = \{e_5, e_2, e_3, e_8\}$. This gives $w(T_1) = 2k-2$ and $w(T_2) = k+2$. This gives a lower bound of $\frac{2k-2}{3k/2} = 4/3 - 4/(3k)$.

reducing the graph, we always take the node with the smallest maximum-weight adjacent edge. This way, when rebuilding the graph, and balancing the trees, we will always start with the maximum-weight edge. When rebuilding we always balance best possible in each step.

Using the strategy **keep-max** with algorithm **split** on the lower bound example from Figure 2 now gives the optimal solution. However, we can show a lower bound of $4/3 - 4$ for algorithm **split** with strategy **keep-max**. For the graph in Figure 3 the optimum solution is to set $T_1 = \{e_1, e_2, e_7, e_8\}$ and $T_2 = \{e_3, e_4, e_5, e_6\}$. This gives $w(T_1) = w(T_2) = 3k/2$. Assume w.l.o.g.. that the algorithm first removes v_5. It will then next remove v_4, and then v_3. In the first rebuilding step we have $T_1 = \{e_1\}$ and $T_2 = \{e_5\}$. In the next step we get $T_1 = \{e_1, e_6\}$ and $T_2 = \{e_5, e_2\}$, in the third step $T_1 = \{e_1, e_6, e_7\}$ and $T_2 = \{e_5, e_2, e_3\}$, and finally $T_1 = \{e_1, e_6, e_7, e_4\}$ and $T_2 = \{e_5, e_2, e_3, e_8\}$. This gives $w(T_1) = 2k-2$ and $w(T_2) = k+2$ this gives a lower bound of $\frac{2k-2}{3k/2} = 4/3 - 4/(3k)$ which is very close to $4/3$ for large k.

We have not yet been able to show that algorithm **split** with strategy **keep-max** gives a better approximation guarantee than the $3/2$ that follows from Theorem 4.1.

7 A Matroid Generalization

In this section we study the further generalization of the partition problem to matroids and give some results on the problem for transversal matroids. We start with a direct generalization of Problem 1.1 to general matroids.

Problem 7.1 *Let $\mathcal{M} = (S, \mathcal{F})$ be a matroid containing two disjoint bases and let ω be a weight function on S. Find a pair of disjoint bases B, B' of $\mathcal{M}$ which minimizes* $\max\{\omega(B), \omega(B')\}$.

It follows from the results proved so far that this problem is NP-hard for graphic matroids (see Section 7.2) and in fact it is even NP-hard when $\mathcal{M}$ is

a uniform matroid (see below).

Problem 7.2 *Let $\mathcal{M} = (S, \mathcal{F})$ be a matroid containing two disjoint bases and let ω be a weight function on S. Find a pair of disjoint bases B, B' of $\mathcal{M}$ which minimizes $\omega(B) + \omega(B')$.*

It follows from Theorem 2.3 that Problem 7.2 is solvable in polynomial time. Furthermore, the observation that we made for graphs in Proposition 1.1 is valid in general.

Proposition 7.1 *Let $\mathcal{M} = (S, \mathcal{F})$ be a matroid containing two disjoint bases and let ω be a weight function on S. If membership of $\mathcal{F}$ can be checked in polynomial time, then there is a 2-approximation algorithm for Problem 7.1 for $\mathcal{M}$.*

PROOF. By Theorem 2.3 we can find a pair of disjoint bases B_1, B_2 in $\mathcal{M}$ which minimize $\omega(B_1) + \omega(B_2)$. Now, for any pair of disjoint bases of $\mathcal{M}$, including the optimal solution (B, B') to Problem 7.1, the sum of the weights of these two bases is at least $\omega(B_1) + \omega(B_2)$, implying that $\max\{\omega(B), \omega(B')\} \geq \frac{\max\{\omega(B_1), \omega(B_2)\}}{2}$, so taking the pair (B_1, B_2) we obtain a 2-approximation of Problem 7.1. □

7.1 Uniform Matroids

The uniform matroid $U_{n,k}$ is the matroid $\mathcal{M} = (S, \mathcal{F})$ such that $|S| = n$ and $\mathcal{F}$ is precisely the collection of subsets of S of size at most k. The problem **Partition**$(2k)$ was defined above. It is easy to see that this problem is equivalent to Problem 7.1 for the uniform matroid $U_{n,k}$, where k is the same as above, n is the number of integers for the instance of **Partition**$(2k)$ and the weight ω on the elements of $U_{n,k}$ is simply the value of the corresponding integers.

The next result shows that (for uniform matroids) it is enough to solve Problem 7.1 on the uniform matroid induced by the $2k$ elements of smallest weights. Note that this set is exactly the union of two disjoint bases whose sum is minimum among all pairs of disjoint bases in $U_{n,k}$. Hence for uniform matroids the minsum reduction preserves at least one optimal solution to Problem 7.1.

Proposition 7.2 *Let S be a set of $r \geq 2k$ non-negative numbers $s_1 \leq s_2 \leq \ldots \leq s_{2k} \leq s_{2k+1} \leq \ldots \leq s_r$, $r \geq 2k$. Let $S_{2k} = \{s_1, s_2, \ldots, s_{2k}\}$. Then every optimal solution to the problem* **Partition**$(2k)$ *for the set S_{2k} is also an optimal solution to the problem* **Partition**$(2k)$ *for the full set S.*

PROOF. For any subset $Z \subseteq S$ we denote by $s(Z)$ the sum $s(Z) = \sum_{s_i \in Z} s_i$. Let X, Y denote an optimal solution to **Partition(**$2k$**)** for S_{2k}, where $s(X) \geq s(Y)$ and suppose that there exists a subset $W \subset S$ such that $|W| = 2k$, $W \neq S_{2k}$, and a partition P, Q of W into two sets of size k where $s(Q) \leq s(P) < s(X)$. Assume furthermore that $W \cap S_{2k}$ is maximum among all such W and that among all such subsets $s(W)$ is as small as possible. If $P \subset X \cup Y$, then by the ordering of the elements of S we have $s(P) \geq s(Q) \geq s(X - P) + s(Y - P)$, but then $(P, S_{2k} - P)$ is a better partition of S_{2k}, contradicting the optimality of (X, Y). Hence we must have $P - (X \cup Y) \neq \emptyset$. If we also have $Q - (X \cup Y) \neq \emptyset$, then let $p \in P - (X \cup Y)$ and $q \in Q - (X \cup Y)$ be arbitrary. Let $u, v \in (X \cup Y) - (P \cup Q)$ be arbitrary distinct elements such that $s(u) \geq s(v)$. By the ordering of S we have $p + q \geq u + v$. If $s(P - p + u) \geq s(Q - q + v)$ we let $P^* = P - p + u$ and $Q^* = Q - q + v$ and otherwise let $P^* = Q - q + v$ and $Q^* = P - p + u$. Now (P^*, Q^*) satisfies that $s(Q^*) \leq s(P^*) < s(X)$ and $W^* = P^* \cup Q^*$ has a larger intersection with S_{2k} than W, contradicting the choice of W. Hence we may assume that $Q \subset X \cup Y$ and $P - (X \cup Y) \neq \emptyset$. Consider any element $z \in S_{2k} - W$. If $s(z) < s(q)$ for some $q \in Q$, then we obtain a better W by swapping z and q, contradiction. Hence for every possible choice of z and q above we have $s(z) \geq s(q)$. Now let $\tilde{P} = S_{2k} - Q$ and observe that $s(X) > s(P) \geq s(\tilde{P})$. Since we also have $s(X) > s(P) \geq s(Q)$ we see that the partition of S_{2k} into Q and $\tilde{P}$ is better than X, Y, a contradiction. □

There exists a fully polynomial time approximation scheme (in short an FPTAS) for **Partition(**$2k$**)**. It is well-known that this is the case for the standard formulation (without requirement on equal size of the sets) of partition problem (see e.g. [4, Section 35.5][7]). This FPTAS can be modified to the case where we require the two sets to be of equal size as follows. The algorithm from [4] works by iteratively computing a "trimmed" list L_i over all possible sums of all subsets of the first i elements that do not exceed the target value (in this case $(\sum_{s \in S_{2k}} s)/2$) from $i = 1$ to $2k$. That the list L_i is trimmed means that as many elements as possible are removed from L_i in such a way that for every removed element y there is an element $z \in L_i$ such that $\frac{y}{1+\delta} \leq z \leq y$. We say that y is represented by z in L_i.

To obtain a FPTAS for our case we associate to each value $z \in L_i$ a list M_z containing the different sizes of subsets of the first i items that sums to z or a value represented by z. The length of M_z is at most i (in fact we can restrict M_z to only contain numbers of size at most k). To each number j in M_z we associate a subset of j items whose values sum to z or a value represented by z. It is easy to verify that keeping this extra modification only gives a blow-up of size $O(k)$ in both time and space. After computing L_{2k} we find the largest value z in L_{2k} with $k \in M_z$ and return the subset associated with k in M_z. Using $\delta = \varepsilon/2n$ for trimming the lists this gives a FPTAS for partition in the

[7] The partition problem is a special case of the subset sum problem.

case where the two sets are required to be of equal size. The proof is omitted since it is a pretty straight-forward modification of the proof from [4].

Now, first applying the reduction from Proposition 7.2 and then the algorithm just described gives the following result.

Proposition 7.3 *There exists a FPTAS for* **Partition**$(2k)$.

7.2 Graphic Matroids

A matroid $\mathcal{M} = (S, \mathcal{F})$ is **graphic** if there exists a graph $G = (V, E)$ with $|E| = |S|$ and there exists a mapping $\phi : S \to E$ such that ϕ is 1-1 and onto and $X \in \mathcal{F}$ if and only if $\phi(X)$ induces an acyclic subgraph of G, that is, $\mathcal{M}$ is isomorphic to the circuit matroid $\mathcal{M}(G)$ of G.

Thus if the rank of $\mathcal{M}$ is $|V| - 1$, then G is connected and spanning trees of G are in 1-1 correspondence to bases of $\mathcal{M}$.

All the results in Sections 3-5 can be rephrased as results about Problem 7.1 for graphic matroids.

7.3 Transversal Matroids

Another well studied class of matroids is the class of transversal matroids. A matroid $\mathcal{M}$ is a transversal matroid if there exists a bipartite graph $\mathcal{B} = (S \cup T, E)$ such that $\mathcal{M} = (S, \mathcal{F})$ where the subsets $X \subseteq S$ that belong to $\mathcal{F}$ are precisely those subsets that can be matched to a subset in T. In other words X is in $\mathcal{F}$ if an only if $\mathcal{B}$ has a matching meeting all vertices of X. We say that the bipartite graph $\mathcal{B}$ **represents** $\mathcal{M}$. Note that $\mathcal{B}$ is generally not unique. For basic properties of transversal matroids we refer the reader to [11].

Transversal matroids are closely related to practical applications and the bases of transversal matroids are also known as **systems of distinct representatives** which have been studied already by Philip Hall. The famous Hall's theorem [2, Theorem 3.11.3] gives a necessary and sufficient condition for the existence of a transversal of a given family of subsets of a set (corresponding to a matching meeting all vertices of T above). By Hall's theorem a system of distinct representatives exists for a family $\mathcal{F}$ of sets if and only if for every sub collection of sets $\mathcal{F}$' from $\mathcal{F}$ the following holds

$$|\bigcup_{i \in \mathcal{F}'} X_i| \geq |\mathcal{F}'|.$$

Since the uniform matroid $U_{n,k}$ can be seen as the transversal matroid represented by a complete bipartite graph $K_{n,k}$ (n vertices in one side and k in the other) it follows that Problem 7.1 is also NP-hard for transversal matroids.

The following observation is a trivial consequence of the definition of a base in a matroid.

Proposition 7.4 *If $\mathcal{M} = (S, \mathcal{F})$ is a transversal matroid, B is a base and $T' \subseteq T$ is a subset of T such that B is matchable to T', then no vertex of $S - B$ is adjacent to any vertex of $T - T'$ in $\mathcal{B}$. In particular, all bases of $\mathcal{M}$ which are disjoint from B are matchable to T' and to no other subset of size $|B|$ in T.*

Theorem 7.1 *Let $\mathcal{M} = (S, \mathcal{F})$ be a transversal matroid and let $\mathcal{B} = (S, T, E)$ be a graph representing $\mathcal{M}$. Suppose B_1, B_2 are disjoint bases of $\mathcal{M}$ such that $S = B_1 \cup B_2$. Let $\omega : S \to \mathcal{R}$ be a weight function on S. In polynomial time we can find a pair of disjoint bases B_h, B_l such that $\omega(B_h) \geq \omega(B_l)$ and*

$$\omega(B_h) - \omega(B_l) \leq \max_{e \in B_h} \omega(e) - \min_{e \in B_l} \omega(e) \tag{21}$$

PROOF. By Proposition 7.4 we may assume that $|T| = |B_1|$, that is, all bases correspond to perfect matchings with respect to T in $\mathcal{B}$. Let M_1 and M_2 be matchings with end vertices $B_1 \cup T$ respectively $B_2 \cup T$ and let the elements of $B_1 = \{x_1, x_2, \ldots, x_t\}, B_2 = \{y_1, y_2, \ldots, y_t\}$, $t = |T|$, be labeled so that x_i and y_i are matched to the same vertex of T by M_1 and M_2. Construct B_h and B_l as follows: Let $H = \emptyset = L$. If $\omega(x_1) \geq \omega(y_1)$ then $H := H + x_1$ and $L := L + y_1$ otherwise $H := H + y_1$ and $L := L + x_1$. In the i-th step add that one of x_i, y_i with the largest weight to L and the other to H and rename H, L (if necessary) so that we always have $\omega(H) \geq \omega(L)$. Finally, after processing all t pairs we let $B_h = H$ and $B_l = L$. It is easy to see that B_h, B_l satisfy (21) (the argument is analogous to that for the case of a 2-vertex in the proof of Theorem 3.1) and it follows from the way we paired the elements that B_h, B_l are bases of $\mathcal{M}$. □

Applying first a min sum reduction and then using Theorem 7.1 and arguments analogous to those used in Section 4 we can show the following. Note that a polynomial independence oracle can be implemented using flows in networks (see [2, page 140]).

Theorem 7.2 *There exists a $\frac{3}{2}$-approximation algorithm for Problem 7.1 in the class of transversal matroids.* [8]

[8] Here we assume that the input is a bipartite graph $\mathcal{B} = (S, T, E)$ representing the transversal matroid and a weight function on S.

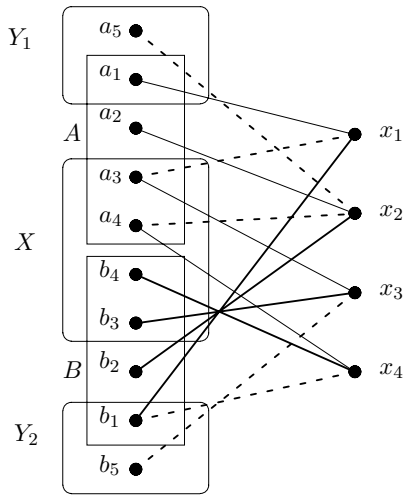

Fig. 4. A bipartite graph representing transversal matroid on 10 elements $\{a_1, a_2, a_3, a_4, a_5, b_1, b_2, b_3, b_4, b_5\}$ The fours bases $A = \{a_1, a_2, a_3, a_4\}, B = \{b_1, b_2, b_3, b_4\}, X = \{a_3, a_4, b_3, b_4\}$ and $Y = Y_1 \cup Y_2 = \{a_1, a_5, b_1, b_5\}$ are shown.

We will now show that the min sum reduction may remove all optimal solutions to Problem 7.1 in the case of transversal matroids.

Theorem 7.3 *Let $\mathcal{M}$ be the transversal matroid represented by the bipartite graph in Figure 4. There exists a weight function ω and some optimal solution (B_1, B_2) to Problem 7.2 on $\mathcal{M}$ so that (B_1, B_2) is an optimal solution to Problem 7.1 on the restriction of $\mathcal{M}$ to $B_1 \cup B_2$ and*

$$\max\{\omega(B_1), \omega(B_2)\} = \frac{7}{6} \max\{\omega(B_h), \omega(B_l)\}, \tag{22}$$

where (B_h, B_l) is an optimal solution to Problem 7.1 on $\mathcal{M}$.

PROOF. Let X, Y, A, B be bases as defined in Figure 4 and let k be an arbitrary positive integer and define ω by

$$\omega(\{a_1, a_2, a_3, a_4, a_5, b_1, b_2, b_3, b_4, b_5\}) = \{2k, 2k, k, 2k, 2k, 0, 2k, k, 2k, 2k\}.$$

Then $\omega(A), \omega(B) = 7k, 5k$ and $\omega(X), \omega(Y) = 6k, 6k$.

First observe that, since every such pair has exactly 8 elements, no pair of disjoint bases have a total weight of less than $12k$ and hence A, B is an optimal solution to Problem 2.3 in $\mathcal{M}$.

We also claim that A, B is an optimal solution to problem 7.1 in the restriction of $\mathcal{M}$ to $A \cup B$. Suppose that there is a better partition B_1, B_2 of $A \cup B$ into disjoint bases of $\mathcal{M}$. Then clearly we must have $\omega(B_i) = 6k$ and this

implies that w.l.o.g. $\{a_3, b_3\} \subset B_1$ and $b_1 \subset B_2$. Since a_3 and b_3 are the only elements in $A \cup B$ which are adjacent to x_3 it is not possible to extend B_2 to a base. Hence A, B is indeed an optimal partition. As $\frac{\omega(A)}{\omega(X)} = \frac{7}{6}$ the theorem follows. □

Note also that by using instead the weight assignment

$$\omega(\{a_1, a_2, a_3, a_4, a_5, b_1, b_2, b_3, b_4, b_5\}) = \{2k{+}1, 2k, 2k{+}1, k, 2k, 0, 2k, 2k{+}1, k, 2k{+}1\}$$

we can obtain $\omega(X) + \omega(Y) = 12k + 4 > 12k + 3 = \omega(A) + \omega(B)$ and still have $\frac{\omega(A)}{\omega(X)}$ roughly $\frac{7}{6}$.

Many transversal matroids are also graphic matroids (see e.g. [11]) so the question is whether an example as above could be obtained from a corresponding example for graphic matroids. However the situation changes if we restrict our attention to transversal matroids with two disjoint bases and simple graphs.

Lemma 7.1 *A transversal matroid $\mathcal{M} = (S, \mathcal{F})$ containing two disjoint bases B_1, B_2 is the graphic matroid $\mathcal{M}(G)$ of some simple graph G only if $S = B_1 \cup B_2$.*

PROOF. It follows from a result of [3] that the graphic matroid of a simple graph G is a transversal matroid if and only if G contains no subdivision of K_4. By a result of [5, Satz 5] every simple graph of minimum degree at least 3 contains a subdivision of K_4. Thus to prove the claim it suffices to observe that bases of $\mathcal{M}$ correspond to bases in $\mathcal{M}(G)$, that is, spanning trees of G. Suppose that $\mathcal{M}$ is the transversal matroid of the simple graph G and that G contains no subgraph of minimum degree greater than 2. Then it is easy to prove by induction that G is exactly the union of two spanning trees. □

If we restrict the problem to transversal matroids consisting only of two disjoint bases there exists a FPTAS for the problem.

Proposition 7.5 *Let $M = (S, \mathcal{F})$ be a transversal matroid and let $\mathcal{B} = (S, T, E)$ be a graph representing $\mathcal{M}$. If B_1, B_2 are disjoint bases of $\mathcal{M}$ such that $S = B_1 \cup B_2$ then there exists a FPTAS for Problem 7.1 on $\mathcal{M}$.*

This FPTAS can be obtained as follows. In the case where S consists of only two disjoint bases, we can view this as another version of **Partition** (compare with the proof of Theorem 7.1), where we have a list of pairs $(x_1, y_1), \ldots, (x_n, y_n)$ and wish to split these pairs into two sets such that for each pair (x_i, y_i), x_i and y_i are put into different sets, minimizing the sum of

the elements in the maximum of the two sets. This is the same version that we used in the proof of Proposition 2.1.

The FPTAS can be obtained by a small modification of the FPTAS for Subset sum from [4, Section 35.5]. Iteratively compute a trimmed list L_i of all possible sums of the first i pairs that do not exceed the target value (in this case $(\sum_{s \in S} s)/2$) as follows. For each element ℓ in L_i we add the two elements $\ell + x_i\hat{\text{E}}$ and $\ell + y_i$ to L_i and thereafter trim the list as described in [4]. Finally, return the largest element in L_n. That this gives a FPTAS follows directly from the proof in [4].

If we can show that the maximum relative error we may make by performing the min sum reduction before solving Problem 7.1 for transversal matroids is α we immediately obtain an $\alpha(1+\varepsilon)$-approximation for Problem 7.1 on transversal matroids by first applying the min sum reduction and then the FPTAS from Proposition 7.5.

8 Remarks and Open Problems

We have observed that the problem Partition($2k$) is equivalent to Problem 7.1 for the uniform matroid $U_{n,k}$ and we generalized the partition problem to several classes of matroids:

For uniform matroids the min sum reduction does not exclude all optimal solutions to Problem 7.1 and there exists a FPTAS.

For transversal matroids we have shown that the min sum reduction can exclude the optimal solutions to the problem, but the relative error is at most $3/2$ (and we have given examples where it is $7/6$). For the case where the matroid only consists of two disjoint bases there exists a FPTAS.

For graphic matroids (Problem 1.1) we have shown that the min sum reduction can exclude the optimal solutions to the problem, but the relative error is again between $7/6$ and $3/2$. Even for the case of 2T-graphs we have not been able to show a better approximation guarantee than $3/2$.

This leaves the following open problems.

- For matroids that are either graphic or transversal matroids what is the maximum relative error we may make by performing the min sum reduction before solving Problem 7.1. Our examples and Theorems 4.1, 7.2 show that the answer lies between $\frac{7}{6}$ and $\frac{3}{2}$.
- Is there a polynomial time approximation scheme for Problem 1.1 on 2T-graphs?

- What is the maximum ratio $\frac{\omega(T_h)+\omega(T_l)}{\omega(T_1)+\omega(T_2)}$ where (T_h, T_l) and (T_1, T_2) are optimal solutions to Problem 1.1 and Problem 1.2 respectively on the same graph? Our examples above show that the fraction can be as high as $\frac{4}{3}$.
- Can we obtain a better approximation guarantee than 3/2 for Problem 7.1 on graphic or transversal matroids?
- We saw in Section 5 that the min sum reduction does not always preserve optimal solutions so an interesting question seems to be whether there is some other reduction from the general case to 2T-graphs which does preserve optimal solutions.

The problem **Partition** is known to be easily solvable, that is, for most instances it is easy to obtain good solutions (see e.g. [7,9]) and the problem has a fully polynomial-time approximation scheme [4, Section 35.5]. Hence one may ask whether this property of being easy to solve well is preserved when we add the further structure requirements to the problem by considering Problem 1.1. We investigated this in [1] and our test results seem to indicate that this is indeed the case.

Recently, after the completion of this paper, van den Heuvel and Thomassé [8] proved a very interesting result on matroids. They proved that a matroid with k disjoint bases and rank r has a cyclic ordering on the kr elements of these bases such that any r consecutive elements of this order form a base. This implies that given two disjoint bases B_1 and B_2 minimizing $\omega(B1) + \omega(B2)$ there exist a cyclic ordering $e_1, e_2, \ldots, e_{2r}$ of the $2r$ elements of these bases such that any r consecutive elements of this order form a base. Let B_i be the base with elements $e_i, e_{i+1}, \ldots, e_{i+r}$ (the indices are modulo $2r$) and let $\delta_i = \omega(B_i) - \omega(B_i + r)$. It is clear that $\delta_i = -\delta_{i+r}$, so there is an index i such that $\delta_i \geq 0$ and $\delta_{i+1} < 0$. Furthermore since for any i we have $\delta_i - \delta_{i+1} = 2(\omega(e_{i+r}) - \omega(e_i))$ it is clear that for some i $|\delta_i| \leq \omega_{MAX} - \omega_{MIN}$, where ω_{MAX} and ω_{MIN} are respectively the minimum and the maximum weigh of an element. So their result implies Theorem 3.1 for arbitrary matroids.

References

[1] J.Bang-Jensen and C. B. Christensen, Heuristics for finding well-balanced pairs of edge-disjoint trees in weighted graphs, *International Journal of Operational Research*, to appear.

[2] J.Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer Verlag London 2001.

[3] J.A.Bondy, Transversal matroids, base-orderable matroids and graphs, *Quarterly J. Math. Oxford 2nd series* **23** (1971) 81-89.

[4] T.H.Cormen, C.E. Leiserson, R. Rivest and C. Stein, *Introduction to Algorithms 2nd edition, MIT Press* (2001).

[5] G.A. Dirac, In abstrakten Graphen vorhandene vollständige 4-Graphen un ihre Unterteilungen, *Mathematische Nachrichten, Akademie-Verlag Berlin* **22** (1960) 61-85.

[6] M.R. Garey and D.S. Johnson, *Computers and Intractability.* W. H. Freeman an d Co, San Francisco, Calif. 1979.

[7] B. Hayes, "The Easiest Hard Problem" A reprint from *American Scientist the magazine of Sigma Xi, the Scientific Research Society* Volume 90, Number 2 March-April, 2002 pages 113-117.

[8] J. van den Heuvel and S. Thomassé, Cyclic orderings of matroids, CDAM Research Report LSE-CDAM-2007-14 (2007).

[9] S. Mertens, "A physicist's approach to number partitioning" *Theoretical Computer Science* **265** (2001) 79–108.

[10] A. Recski: *Matroid theory and its applications in electric network theory and in statics* (Algorithms and Combinatorics, Vol. 6) Springer Verlag, Berlin New York and Akadémiai Kiadó, Budapest, 1989.

[11] D.J.A.Welsh, *Matroid Theory*, Academic Press 1976.

[12] D.B.West, *Introduction to Graph Theory*, Prentice Hall NJ, 1996.

Appendix

Details of the FPTAS for two disjoint bases of transversal matroids

The following description is based on [4].

The lists are trimmed as follows. To trim a list L by δ, remove as many elements from L as possible, in such a way that for every element y that was removed, there is an element z still in the list that approximates y, that is

$$\frac{y}{1+\delta} \leq z \leq y \ .$$

This can be done by going trough L in sorted order and keeping an element y only if y is greater than $(1+\delta)$ times the last element we kept. After trimming the list we remove all elements larger than the target value from L.

To obtain a $(1+\varepsilon)$-approximation we will trim the lists with $\varepsilon/2n$.

Proof of Proposition 5

Let y^* be the optimal value. The value z^* returned is the sum of some subset of S, but not necessarily the sum of k elements. But it is representing the sum of k elements. Let z' denote the actual sum of k elements that z^* is representing. We know that $z' \leq w(S)/2$. It can be shown by induction that

$$\frac{y^*}{(1+\varepsilon/2n)^n} \leq z^* \leq y^* \ .$$

We have $y^* \geq z' \geq z^*$ due to the optimality of y^* and thus

$$y^* \leq (1+\varepsilon/2n)^n z^* \leq (1+\varepsilon/2n)^n z' \leq (1+\varepsilon) z' \ .$$