Operations Research Letters 41 (2013) 321–324

Contents lists available at SciVerse ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

Approximating max–min weighted T-joins

Satoru Iwata [a], R. Ravi [b,*]

[a] Research Institute for Mathematical Sciences, Kyoto University, Kyoto 606-8502, Japan

[b] Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA

ARTICLE INFO

Article history:
Received 3 January 2013
Received in revised form
11 March 2013
Accepted 12 March 2013
Available online 22 March 2013

Keywords:
T-joins
Max–min optimization
Bottleneck problem

ABSTRACT

Given an undirected graph with nonnegative edge weights, the max–min weighted T-join problem is to find an even cardinality vertex subset T such that the minimum weight T-join for this set is maximum. The problem is NP-hard even on a cycle but permits a simple exact solution on trees. We present a 2/3-approximation algorithm based on a natural cut packing upper bound by using an LP relaxation and uncrossing, and relating it to the T-join problem using duality.

1. Introduction

The T-join problem is an important generalization of matching problems. Given an undirected graph with an even subset T of its vertices, a T-join is a subgraph in which the subset of vertices with odd degree is exactly T. Given edge weights, the weighted T-join problem is to find a T-join of minimum weight. With nonnegative edge weights, the problem can be reduced to finding a minimum weight perfect matching on the metric completion of the vertices in T.

In this paper, we consider the *max–min weighted T-join* problem: Given an undirected graph $G = (V, E)$ with nonnegative weights w on the edges, find an even cardinality subset T of vertices such that the minimum weight T-join for this set is maximum. The unweighted case of the problem when all weights are either unit or infinity has been well studied [6]: the main result there characterizes the size of the max–min T-join by relating it to the minimum number of odd ears in an ear decomposition, and also to the minimum number of edges in the graph whose contraction makes it factor critical. The resulting min–max equality also gives a polynomial-time algorithm for the max–min T-join problem in unweighted (or unit-weighted) graphs.

While the question of the weighted case was left open by Frank [6], Ageev [1] settled the complexity and showed it to be NP-hard. The version that he showed NP-hard is the following: Given an undirected graph with nonnegative weights on the edges, define a join to be a subset of edges such that the *number* of edges in the join from any cycle is at most half the cardinality of the cycle; Ageev showed that finding the join of maximum weight is NP-hard. However, the max–min weighted T-join problem we consider is not the same as the weighted max join of Ageev: In particular, for a subset of (weighted) edges to be a minimum weight T-join for some set of vertices T, it must be the case that for any cycle, the T-join must intersect the cycle in at most half the *weight* of the cycle; otherwise, by taking the symmetric difference of the edges in the T-join and this cycle, its weight can be reduced, contradicting the fact that it was a minimum weight T-join. Thus for the max–min weighted T-join problem we study, a feasible minimum-weight T-join is a subset of edges such that the *weight* of the edges in the T-join from any cycle is at most half the total weight of the cycle. (One can refer to the weighted version that Ageev showed NP-hard simply as the max weighted join problem, since the solution to his problem does not necessarily correspond to any even cardinality vertex set T.)

Our results. We observe that the max–min weighted T-join problem has an easy solution in a tree. We then show a $\frac{2}{3}$-approximation algorithm for the max–min weighted T-join problem using a simple cut packing upper bound. We close with some extensions and open problems.

Related work. Max–min optimization problems over the choice of the set of demand elements (such as the set of vertices to be included in T in our case) have been studied before in the context of robust optimization under demand uncertainty [9]. The cardinality version they study is of the form "Which k demand elements result in the corresponding minimization problem having the largest value?" For example, in a capacitated graph, find the k

* Corresponding author. Tel.: +1 4122683694.
E-mail addresses: iwata@kurims.kyoto-u.ac.jp (S. Iwata), ravi@andrew.cmu.edu, ravi@cmu.edu (R. Ravi).

0167-6377/$ – see front matter
http://dx.doi.org/10.1016/j.orl.2013.03.004

sources whose separation from a given sink costs the most. This latter problem is an instance of monotone submodular function maximization under a cardinality constraint and has a $(1 - \frac{1}{e})$-approximation algorithm [13,5]. However, problems that do not fit in the submodular framework have also been analyzed in this cardinality model such as set cover [4], Steiner forest and multicut [9]; these papers derive approximation algorithms whose performance nearly matches that of the corresponding minimization problem. Our work has the same flavor but imposes no cardinality constraint on the set of demand vertices (T-vertices) chosen except that it is even.

Max–min optimization has also been studied widely in other less closely related areas of study. The max–min objective arises in bottleneck optimization problems [10], where the minimum weight element in the solution is sought to be maximized. Max–min problems arise in the area of robust optimization under data uncertainty, where the minimized object is fixed (such as a spanning tree) and the max is taken over the various data scenarios (e.g., of edge costs) [15]. Yet another stream of work uses max–min objectives as a condition of fairness in choosing an efficient solution for multiple criteria (such as the load on various edges in a routing problem)—see, e.g., [12]. Our work differs from these in that the maximization is over the choice of demand elements to include in the optimization problem, rather than over the various cost scenarios or over alternate solutions of the same underlying optimization problem with fixed demands.

The max–min weighted T-join problem is distantly related to the analysis of Christofides' $\frac{3}{2}$-approximation algorithm for the metric version of the symmetric traveling salesperson problem [2]. This method finds a connected Eulerian subgraph which is then shortcut by using an Euler tour with no increase in weight due to the metric condition. To obtain the Eulerian subgraph, the method first finds a minimum weight spanning tree (MST) and then finds a minimum-weight T-join of the odd degree vertices in the MST, and uses their union as the Eulerian subgraph. The max–min weighted T-join problem characterizes the worst set of vertices that could arise as T in this method, and hence may prove useful in analysis of adaptations of Christofides' method.

1.1. Optimal solution on trees

Proposition 1.1. *For a weighted tree, defining T to be the odd degree vertices in the tree results in the minimum weight T-join being the whole tree.*

Recall that for a given subset T of even cardinality, a T-cut is a cut in the graph where each shore has an odd number of vertices in T. Given the parity of the degrees in a T-join, any such T-cut must have an odd number of edges (and hence at least one edge) in any T-join for this set T.

We will check that defining the odd degree vertices of a tree H to be the set T obeys the above property. For any tree edge e, observe that each component of $H \setminus \{e\}$ has an odd number of odd degree vertices. Otherwise, the parity of the number of edges leaving such a set (which is the unique edge e and hence is odd) will be even. Thus, every tree edge defines a T-cut when T is the set of odd-degree vertices in H. As argued above, every cut corresponding to a tree edge must be crossed by any T-join. Thus the whole tree H is itself a minimum T-join for this definition of T. Since trivially one cannot obtain a T-join that is larger than the weight of the whole tree for any T, this also shows that this is an optimal solution to the max–min weighted T-join problem on the tree.

1.2. NP-hardness

Sebő [14] showed that the max–min weighted T-join problem is weakly NP-hard even on a cycle by a reduction from the PARTITION problem. Let $a_1, \ldots, a_n$ be an instance of PARTITION. To reduce this to the weighted max–min T-join problem, consider a cycle of size n with edge-weights $a_1, \ldots, a_n$ in an arbitrary order. It is now easy to check that the maximum weight of a T-join problem on this instance is $\frac{a_1+\cdots+a_n}{2}$ if and only if the given instance of PARTITION is feasible, and is strictly smaller otherwise. This follows in particular from the condition alluded above that any feasible T-join to our problem is a subset of edges such that the *weight* of the edges in the T-join from this cycle is at most half the total weight of the cycle.

2. The $\frac{2}{3}$-approximation algorithm

Let $G = (V, E)$ be a graph with vertex set V and edge set E, where each edge has a *nonnegative* weight $w(\cdot)$. In this section, we present an approximation algorithm for the max–min weighted T-join problem on G. The algorithm is based on an upper bound obtained by linear programming duality.

2.1. A linear programming upper bound

For a vertex subset $S \subseteq V$, we denote by δS the set of edges between S and $V \setminus S$. For a vertex subset $T \subseteq V$ of even cardinality, δS is called a T-cut if $|S \cap T|$ is odd. Let $\mathcal{Q}_T$ denote the set of vertex subsets that determine a T-cut and do not contain a specified vertex $r \in V$, i.e., $\mathcal{Q}_T = \{S \mid r \notin S \subseteq V,\ |S \cap T| \text{ odd}\}$. Edmonds and Johnson [3] have shown that the following linear program, for nonnegative weights $w(\cdot)$, has an integral optimal solution, which corresponds to a minimum weight T-join.

$$\text{(TJ)} \quad \text{Minimize} \sum_{e \in E} w(e)x(e)$$

$$\text{subject to} \sum_{e \in \delta S} x(e) \geq 1, \quad \forall S \in \mathcal{Q}_T,$$

$$x(e) \geq 0, \quad \forall e \in E.$$

The dual linear program is as follows.

$$\text{(TCP)} \quad \text{Maximize} \sum_{S \in \mathcal{Q}_T} y_S$$

$$\text{subject to} \sum_{S: e \in \delta S} y_S \leq w(e), \quad \forall e \in E,$$

$$y_S \geq 0, \quad \forall S \in \mathcal{Q}_T.$$

We now provide an upper bound on the weight of a max–min T-join over all possible sets T. To do this, we consider the following linear program.

$$\text{(CP)} \quad \text{Maximize} \sum_{S \subseteq V \setminus \{r\}} y_S \tag{1}$$

$$\text{subject to} \sum_{S: e \in \delta S} y_S \leq w(e), \quad \forall e \in E, \tag{2}$$

$$y_S \geq 0, \quad \forall S \subseteq V \setminus \{r\}. \tag{3}$$

Note that (CP) and (TCP) differ only in the set of y-variables in them: The former contains one for every subset while the latter has variables only for the odd T-sets. The linear program (CP) finds the maximum fractional packing of cuts in the weighted edges of the given graph. To see why this is an upper bound on the value of any T-join, observe that an arbitrary feasible solution of (TCP) is feasible to (CP) for any choice of T. Therefore, the optimal value of (CP) is greater than or equal to the optimal value of (TCP), and by the strong duality, to the optimal value of (TJ), which is equal to the minimum T-join weight by the integrality [3]. Thus the optimal value of the linear program (CP) provides an upper bound on the maximum value of a minimum T-join weight. Let z_{CP} denote the optimal value of (CP). We summarize this discussion below.

Proposition 2.1. *For any graph with nonnegative edge weights, the max–min weighted T-join has weight at most the value z_{CP} for this graph.*

The linear program (CP) has an exponential number of variables. However, the separation problem of the dual linear program of (CP) is efficiently solvable as it reduces to a min-cut problem. Therefore, one can solve (CP) in polynomial time via the ellipsoid method [8]. Alternatively, one can make use of a compact extended formulation for the dual of (CP).

We now argue that (CP) admits an optimum cut packing y^* such that $\Psi = \{S \mid y^*_S > 0\}$ forms a *laminar* family. This requires that any pair $S_i, S_j \in \Psi$ must obey that either $S_i \subset S_j, S_j \subset S_i$ or $S_i \cap S_j = \emptyset$ (recall that all sets S in Ψ do not contain r). This follows from a simple uncrossing argument: Whenever we have two sets S_i and S_j that are given positive value by y^* and cross, i.e., $S_i \cap S_j$, $S_i \setminus S_j$, $S_j \setminus S_i$ are all nonempty, then we can change the value of y^* as follows: Let $\epsilon = \min(y^*_{S_i}, y^*_{S_j})$. Reduce the y^* value of both S_i and S_j by ϵ and increase by ϵ the y^* values of the cuts defined by the shores $S_i \cap S_j$ and $S_i \cup S_i$. It is easy to verify from the submodularity of the cut function that this change does not result in overpacking the edge weights as constrained by (2). Moreover, it is not hard to see that the change increases the value of $\sum_S y^*_S \cdot |S|^2$ [11]. Thus choosing an optimal solution y^* that maximizes $\sum_S y^*_S \cdot |S|^2$ will necessarily give a solution with no pair of crossing sets. Such a cross-free family of sets is laminar as claimed.

2.2. Constructing an approximate solution

In this section, we show how to obtain an even cardinality vertex subset T for which the minimum T-join weight is at least $\frac{2}{3}$ of the upper bound z_{CP}.

Note that a triangle with unit weights shows that the gap between the above upper bound and a max–min weighted T-join can be as low as $\frac{2}{3}$. In fact, the maximum cut packing has value $\frac{3}{2}$, while a max–min weighted T-join consists of a single edge with T being the set of end-vertices of this edge. By connecting many such triangles at a common vertex, the example generalizes for any large number of vertices. We show that this bound can be realized by an algorithm.

Our algorithm starts by solving the linear program (CP) and using the solution to obtain an optimal uncrossed solution. First, we observe that even though the number of variables in (CP) is exponential, the number of nontrivial constraints is polynomial (as many as the number of edges) and hence at any extreme point solution, the number of nonzero variables is also at most the number of edges. This extreme point solution to (CP) will in general be made of crossing cuts, but they can be uncrossed in polynomial time using the methods in [11]. This leads to an optimal uncrossed solution y^* such that the family Ψ of shores with positive y^*_S values is a laminar family.

We then naturally represent this family Ψ as a tree H whose root is r and leaves are the vertices of $V \setminus \{r\}$. The tree has an internal node for every set $S \in \Psi$. A leaf v has $S \in \Psi$ as its parent if S is the smallest member in Ψ that contains v. Note that if the singleton cut $\{v\}$ has positive y^*-value, we make a new internal node for the set $\{v\}$ even though there is a separate leaf node for this vertex v as well in H. Every internal node S has as its parent the internal node corresponding to the smallest set $R \in \Psi$ such that $S \subsetneq R$. All the maximal sets in Ψ and the leaves that do not belong to any member of Ψ are assigned r as their parent. (Note that r is a vertex of the graph G that is not represented in the leaves and is not in the shore of any of the cuts representing the internal nodes of H.)

The problem of defining T can now be rephrased in the tree H, where we think of assigning nodes of V for inclusion in the set T. First, we use the fact that for any assignment of the leaves to T, any T-cut must be crossed by any T-join. Thus an assignment of the leaves to T to maximize the number of cuts in the given packing Ψ that are T-cuts gives a lower bound on the size of the resulting T-join. We denote an internal node of H corresponding to a subset S as a node having weight y^*_S. Thus the problem of defining T is now one of assigning the leaves of H to T such that we maximize the total weight of the internal nodes that have an odd number of leaves in T in the subtree under them.

The algorithm below defines a way to choose the subset of leaves so as to get a $\frac{2}{3}$ fraction of the total node weight of the tree. In particular, we show the following.

Proposition 2.2. *Given an undirected tree H with nonnegative weights $g(\cdot)$ on its internal nodes, one can find in polynomial time a subset T of the leaves of H such that the total weight of internal nodes whose subtrees contain an odd number of leaves in T is at least $\frac{2}{3}$ of that of the total node weight of the tree.*

Proof. First, we observe that the tree can be assumed to be binary (i.e., each internal node has at most two children) without loss of generality. This is because high degree nodes can be replaced by any binary subtree rooted at the high-degree node and having the children of the high-degree node as the leaves; all new internal nodes are assigned zero weight. It is easy to check that under any assignment of the leaves to T, the binary version of the original tree accumulates the same node weight of internal nodes with an odd number of T nodes under them.

Next, we argue that we can find subsets T_1, T_2, T_3 (not necessarily disjoint) of the leaf nodes of the tree H such that every internal node of the tree has an odd number of leaf descendants in exactly two of the three sets T_1, T_2, T_3. We show how we can construct these sets T_1, T_2, T_3 recursively as we work our way bottom up in the tree.

At each leaf node l, two of the three sets T^l_1, T^l_2, T^l_3 are the singleton $\{l\}$ and the other one is empty. For an internal node s with only one child l, we assign $T^s_1 = T^l_1$, $T^s_2 = T^l_2$, $T^s_3 = T^l_3$. At an internal node s with two children p and q, we show how to compose the sets T^p_1, T^p_2, T^p_3 and T^q_1, T^q_2, T^q_3 to get the sets T^s_1, T^s_2, T^s_3. Note that T^p_* is a subset of the leaves in the subtree rooted in p, T^q_* is a subset of the leaves in the subtree rooted at q and hence T^s_* will be a subset of the leaves in the union of these two subtrees. Assume without loss of generality that the nodes p and q have an odd number of leaf nodes in the sets T^p_1, T^p_2 and T^q_1, T^q_2, respectively. Now consider the following assignment:

$$T^s_1 = T^p_2 \cup T^q_3,$$
$$T^s_2 = T^p_3 \cup T^q_1,$$
$$T^s_3 = T^p_1 \cup T^q_2.$$

Since T^p_2 has an odd number of leaves and T^q_3 has an even number of leaves, their union T^s_1 has an odd number of leaves. Similarly, T^s_2 has an odd number of leaves while T^s_3 has an even number of leaves as required (see Fig. 1 for an example). Finally, we apply the same assignment rule to the root r and adopt $T_1 = T^r_1$, $T_2 = T^r_2$, $T_3 = T^r_3$.

Since every internal node has an odd number of leaves in exactly two out of T_1, T_2, T_3, if we denote by L_s the set of leaves of H in the subtree rooted at s, and $\mathrm{Odd}(L_s \cap T_i)$ to be the indicator variable for whether the subtree rooted at s has an odd number of leaves in the set T_i, we get

$$\sum_{i=1}^{3} \sum_{s \in I} g(s) \cdot \mathrm{Odd}(L_s \cap T_i) = \sum_{s \in I} 2g(s),$$

where I denotes the set of internal nodes. Therefore, one of the three sets T_i must satisfy that $\sum_{s \in I} g(s) \cdot \mathrm{Odd}(L_s \cap T_i) \geq \frac{2}{3} \sum_{s \in I} g(s)$, completing the proof. □

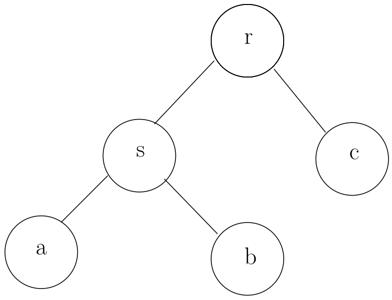

Fig. 1. In the above example, if we start with $T_1^a = \{a\}, T_2^a = \{a\}, T_3^a = \emptyset, T_1^b = \{b\}, T_2^b = \{b\}, T_3^b = \emptyset$, and $T_1^c = \{c\}, T_2^c = \{c\}, T_3^c = \emptyset$, we get $T_1^s = \{a\}, T_2^s = \{b\}, T_3^s = \{a, b\}$ and furthermore $T_1^r = \{b\}, T_2^r = \{a, b, c\}, T_3^r = \{a, c\}$.

Let T be the set T_i chosen at the end of the proof of Proposition 2.2. If T contains an odd number of vertices, then we add r to T. Thus we obtain an even sized subset $T \subseteq V$ that satisfies

$$\sum_{S \in \mathcal{Q}_T} y_S^* \geq \frac{2}{3} \sum_{S \subseteq V \setminus \{v\}} y_S^*.$$

Restricting y^* on $\mathcal{Q}_T$ yields a feasible solution to (TCP). This implies that the minimum weight T-join is at least 2/3 of the cut packing upper bound z_{CP}. Combining this with Proposition 2.1, we have obtained a 2/3-approximation algorithm for the max–min weighted T-join problem.

3. Extensions and open problems

3.1. Max–min point-to-point connection

Given an undirected graph with nonnegative weights on the edges, and two disjoint sets S (sources) and D (destinations) of equal size, an S–D connector is a subgraph in which every connected component has an equal number of sources and destinations. The S–D connector problem has been well studied. Goemans and Williamson [7] give a primal–dual 2-approximation algorithm for the problem. The primal is a cut-covering formulation requiring that every cut with an unequal number of S and D vertices on either shore must be crossed; the dual is thus a maximum packing of such unbalanced cuts (that have different numbers of S and D vertices in either shore). The primal–dual approximation algorithm guarantees that for any choice of S and D, the minimum weight of an S–D connector is at most twice the maximum weight packing of S–D unbalanced cuts.

Our result extends to the max–min version of the problem where the goal is to choose two disjoint subsets of vertices of equal size as sources S and destinations D (i.e., identify $S, D \subseteq V$ such that $S \cap D = \emptyset$ and $|S| = |D|$) so that the minimum weight of a set of S–D connector is maximum.

Note that any packing of unbalanced S–D cuts for any choice of S and D is feasible for (CP), and hence the maximum cut packing objective value of (CP), z_{CP}, is an upper bound on the dual cut packing value for any S–D connector. We use the same algorithm as for the max–min weighted T-join problem to find an even sized subset T, and use any bipartition of this set T into two parts of equal size as the sets S and D for this problem. Note that any T-cut has an odd number of vertices in T and hence is necessarily unbalanced for any such S and D. Thus our choice of the sets S, D results in having an unbalanced cut packing of weight at least $\frac{2}{3} z_{\text{CP}}$.

For any source–destination pair S and D, let $w(\text{Conn}(S, D))$ and $w(\text{Pack}(S, D))$ denote the minimum weight of the integral S–D connector and the optimal value of its fractional linear programming dual respectively. Let $(\tilde{S}, \tilde{D})$ be the pair found by our algorithm and (S^*, D^*) be a pair that maximizes the minimum weight of the S–D connector. Note that $w(\text{Pack}(S^*, D^*)) \leq z_{\text{CP}}$ and that $w(\text{Conn}(S, D)) \leq 2w(\text{Pack}(S, D))$ for any sets S, D using [7]. We then have

$$\begin{aligned} w(\text{Conn}(\tilde{S}, \tilde{D})) &\geq w(\text{Pack}(\tilde{S}, \tilde{D})) \geq \frac{2}{3} z_{\text{CP}} \\ &\geq \frac{2}{3} w(\text{Pack}(S^*, D^*)) \geq \frac{2}{3} \cdot \frac{1}{2} w(\text{Conn}(S^*, D^*)). \end{aligned}$$

Thus we have demonstrated sets S, D that give a $\frac{1}{3}$ approximation for the max–min weighted S–D connector problem.

3.2. Open questions

An alternate way to define a max–min T-join is as the subset of edges of maximum total weight such that negating the sign of the weights of these edges leaves a graph where there are no negative weight cycles (a conservative weighting). Observe that taking the odd degree vertices of this subgraph whose weights have been negated defines a set T for which this subset is a minimum weight T-join. It is interesting to investigate if this alternate characterization can lead to improved algorithms for the max–min weighted T-join problem for special classes of graphs such as series–parallel graphs.

What is the complexity of the max–min size-k-matching problem? Given a complete undirected graph with nonnegative edge costs, the goal in this problem is to find a subset of $2k$ vertices such that the minimum cost perfect matching among these vertices is maximum. The metric version of this problem is also interesting as a special case.

Acknowledgment

Ravi gratefully acknowledges the support of RIMS, Kyoto University, where he was hosted as a visiting professor during the Spring of 2011.

References

[1] A.A. Ageev, Complexity of finding a join of maximum weight, Discrete Appl. Math. 114 (1–3) (2001) 3–7.
[2] N. Christofides, Worst-case analysis of a new heuristic for the travelling salesman problem, Tech. Rep. 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.
[3] J. Edmonds, E.L. Johnson, Matchings, Euler tours, and the Chinese postman problem, Math. Program. 5 (1973) 88–124.
[4] U. Feige, K. Jain, M. Mahdian, V. Mirrokni, Robust combinatorial optimization with exponential scenarios, in: IPCO, 2007, pp. 439–453.
[5] M.L. Fisher, G.L. Nemhauser, L.A. Wolsey, An analysis of approximations for maximizing submodular set functions II, Math. Program. Stud. 8 (1978) 73–87.
[6] A. Frank, Conservative weightings and ear-decompositions of graphs, Combinatorica 13 (1) (1993) 65–81.
[7] M.X. Goemans, D.P. Williamson, A general approximation technique for constrained forest problems, SIAM J. Comput. 24 (2) (1995) 296–317.
[8] M. Grötschel, L. Lovász, A. Schrijver, Geometric Algorithms and Combinatorial Optimization, Springer-Verlag, 1988.
[9] A. Gupta, V. Nagarajan, R. Ravi, Thresholded covering algorithms for robust and max–min optimization, in: S. Abramsky, C. Gavoille, C. Kirchner, F.M. auf der Heide, P.G. Spirakis (Eds.), ICALP (1), in: Lecture Notes in Computer Science, vol. 6198, Springer, 2010, pp. 262–274.
[10] D.S. Hochbaum, D.B. Shmoys, A unified approach to approximation algorithms for bottleneck problems, J. ACM 33 (3) (1986) 533–550.
[11] C.A.J. Hurkens, L. Lovász, A. Schrijver, E. Tardos, How to tidy up your set-system? in: Combinatorics, in: Colloquia Mathematica Societies Janos Bolyai, vol. 52, North-Holland, 1988, pp. 309–314. URL: http://oai.cwi.nl/oai/asset/10169/10169A.pdf.
[12] J. Kleinberg, Y. Rabani, E. Tardos, Fairness in routing and load balancing, J. Comput. System Sci. (1999) 568–578.
[13] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functions I, Math. Program. 14 (1978) 265–294. http://dx.doi.org/10.1007/BF01588971.
[14] A. Sebő, Max–min weighted T-join is NP-hard on a cycle, June 2011, personal communication.
[15] G. Yu, Min-max optimization of several classical discrete optimization problems, J. Optim. Theory Appl. 98 (1998) 221–242. http://dx.doi.org/10.1023/A:1022601301102.