An Improved Approximation Bound for Spanning Star Forest and Color Saving*

Stavros Athanassopoulos, Ioannis Caragiannis, Christos Kaklamanis, and Maria Kyropoulou

Research Academic Computer Technology Institute and
Department of Computer Engineering and Informatics
University of Patras, 26500 Rio, Greece

Abstract. We present a simple algorithm for the maximum spanning star forest problem. We take advantage of the fact that the problem is a special case of complementary set cover and we adapt an algorithm of Duh and Fürer in order to solve it. We prove that this algorithm computes $193/240 \approx 0.804$-approximate spanning star forests; this result improves a previous lower bound of 0.71 by Chen et al. Although the algorithm is purely combinatorial, our analysis defines a linear program that uses a parameter f and which is feasible for values of the parameter f not smaller than the approximation ratio of the algorithm. The analysis is tight and, interestingly, it also applies to complementary versions of set cover such as color saving; it yields the same approximation guarantee of 193/240 that marginally improves the previously known upper bound of Duh and Fürer. We also show that, in general, a natural class of local search algorithms do not provide better than 1/2-approximate spanning star forests.

1 Introduction

We consider the combinatorial problem of decomposing a graph into a forest of stars that span the nodes of the graph. A *star* is a tree of diameter at most two. It consists of a node designated as *center* and (possibly) of *leaves*, i.e., nodes which are connected through edges to the center. The objective of the problem which is known as the *maximum spanning star forest problem* (SSF) is to compute a star decomposition in which the number of leaves (to be considered as the benefit) is maximized. Equivalently, the number of stars has to be minimized. This is equivalent to the *minimum dominating set* problem in which the objective is to compute a minimum-size set of nodes (the dominating set) in the graph so that each node is either part of the dominating set or adjacent to some node in it. The nodes in the dominating set correspond to centers of stars with leaves the nodes outside the dominating set.

* This work is partially supported by the European Union under IST FET Integrated Project FP6-015964 AEOLUS, by the General Secretariat for Research and Technology of the Greek Ministry of Development under programme PENED, and by a "Caratheodory" basic research grant from the University of Patras.

R. Královič and D. Niwiński (Eds.): MFCS 2009, LNCS 5734, pp. 90–101, 2009.

SSF has several applications in the problem of aligning multiple genomic sequences [22], in the comparison of phylogenetic trees [5], and in the diversity problem in the automobile industry [1]. As a combinatorial optimization problem, SSF has been considered in [8,22]. Nguyen et al. [22] prove that the problem is APX-hard by presenting an explicit inapproximability bound of 259/260 and present a combinatorial 0.6-approximation algorithm. Optimal polynomial-time algorithms are presented for special graph classes such as planar graphs and trees. Chen et al. [8] present a better algorithm with approximation ratio 0.71. The main idea is to run both the algorithm of [22], as well as an algorithm that is based on linear programming and randomized rounding and, then, take the best solution. The linear programming relaxation used in [22] has an integrality gap of (at most) 3/4. Interesting generalizations include node-weighted and edge-weighted versions of SSF. [8,22] present approximation algorithms and APX-hardness results for these problems as well. Stronger inapproximability results for these problems recently appeared in [7].

The minimum dominating set problem is a special case of *set cover*. In set cover, we are given a collection $\mathcal{S}$ of sets over a set U of n elements and the objective is to select a minimum-size subcollection $\mathcal{T}$ of sets that contain all elements. The problem is well known to be hard to approximate within a logarithmic factor [13] while the greedy algorithm that iteratively includes in the cover the set that contains the maximum number of uncovered elements achieves a matching upper bound of H_n [19,21]. The same algorithm has an approximation ratio of H_k when the collection is closed under subsets and each set contains at most k elements; this special case of the problem is known as k-set cover. Slightly better results are also known for k-set cover [2,11,14,20]. The most interesting idea that yields these improvements is *semi-local optimization* [11] which applies to 3-set cover. Semi-local optimization is a local search algorithm. The main idea behind it is to start with an empty 3-set covering and augment it by performing local improvements. Note that once the (disjoint) sets of size 3 have been selected, computing the minimum number of sets of size 2 and 1 in order to complete the cover can be done in polynomial time by a maximum matching computation. So, a semi-local (s,t)-improvement step for 3-set cover consists of (i) the deletion of up to t sets of size 3 from the current covering and (ii) the insertion of up to s disjoint sets of size 3 and the minimum number of necessary sets of smaller size that complete the cover, so that the number of sets in the cover decreases; in case of ties, covers with fewer sets of size 1 are preferable. The analysis of [11] shows that the best choice of the parameters (s,t) is $(2,1)$ and that, somehow surprisingly, larger values for these parameters do not yield any further improvements.

As the complement of minimum dominating set, SSF is a special case of *complementary set cover* (in a sense that is explained in detail in Section 2). In complementary set cover, the objective is to maximize the quantity $n - |\mathcal{T}|$. The approximation ratio of a solution for complementary set cover can be thought of as comparing the "distance" of a solution for set cover from the worst possible solution (i.e., the one that uses n sets to cover the elements) to the distance of the

best solution of the worst possible one. This yields an alternative performance measure for the analysis of approximation algorithms; such measures have been considered for many combinatorial optimization problems in the context of differential approximation algorithms [10,9] or z-approximations [16] (see also [3,4]). Duh and Fürer [11] also consider the application of their algorithm on instances of complementary set cover in which the collection of sets is not given explicitly. Among them, the most interesting case is when the elements correspond to the nodes of a graph and the sets in the collection are all the independent sets of the graph. In this case, set cover is equivalent to graph coloring (i.e., the problem of coloring the nodes of a graph so that no two adjacent nodes are assigned the same color and the number of colors is minimized). *Complementary graph coloring* is also known as *color saving*.

In this paper, we take advantage of the fact that SSF is a complementary set cover problem in which the collection of sets is closed under subsets and we use the approximation algorithm of Duh and Fürer [11] to solve its instances (Section 2). We obtain an approximation ratio of $193/240 \approx 0.804$ improving the previously best known bound of 0.71 from [8] and beating the integrality gap of the linear programming relaxation used in [8]. Our analysis is tight, it does not exploit the particular structure of SSF, and essentially holds for the more general complementary set cover and color saving problems as well (see Section 3). The result of [11] is a lower bound of $289/360 \approx 0.803$ on the approximation ratio of the same algorithm on general instances of color saving (and complementary set cover). Although our improvement on the approximation ratio is marginal, our proof technique is very interesting and, conceptually, it could be applicable to other contexts. The proof in [11] is based on a detailed accounting of the performance of the algorithm by a case analysis of the different possible ways the elements of the sets in the optimal solution are covered by the algorithm; among the several different cases, only a few are presented in that paper. Our proof is much different in spirit, it is simpler, and does not require any case analysis; it is based on proving feasibility of a linear program. We note that analysis of purely combinatorial algorithms using linear programs whose objective value reveals the approximation factor (in a different way than in the current paper) has also been used for k-set cover [2], wavelength management in optical networks [6], and facility location [18]. The definition of the linear program in this paper follows the terminology of [2] but the resulting linear program is significantly smaller and our analysis is different. In particular, we show that if the algorithm obtains an at most f-approximate solution for some instance, then an appropriately defined linear program that has f as a parameter and its constraints capture the properties of the instance and the way the algorithm is applied on it is feasible. This is stated as a *parameterized LP Lemma*. So, the lower bound on the approximation ratio follows by proving that the corresponding linear program is infeasible for values of f smaller than $193/240$.

Furthermore, motivated by well-known set packing heuristics [14,17], we consider a natural family of local search algorithms for SSF. Such an algorithm starts with an initial spanning star forest and repeatedly performs local improvements

until this is not possible any more. The resulting solution is a local optimum for the particular algorithm and the question is whether local optima are efficient solutions for SSF as well. We prove that this is not the case for any local search algorithm that belongs to the family and runs in polynomial time. In the proof, we construct almost 1/2-approximate spanning star forests on appropriately defined graphs that are local optima for such algorithms (Section 4).

2 Algorithm Description

In this section, we describe how to exploit the relation of the problem to set cover in order to solve it using a well-known algorithm of Duh and Fürer [11]. We first transform an SSF instance consisting of a graph $G = (V, E)$ into an instance $(V, \mathcal{S})$ of set cover over the nodes. The collection $\mathcal{S}$ consists of *star sets*. A star set is a set of nodes with a common neighbor (assuming that a node neighbors on itself). We distinguish between two types of star sets: A and B. The nodes of a star set s of type A have a node in s as a common neighbor; all other star sets are of type B. Observe that the collection $\mathcal{S}$ that is defined in this way is closed under subsets. We use the term star set cover in order to refer to the particular set cover instance and the term i-star set cover when the collection consists of star sets of size at most i.

We use the algorithm of Duh and Fürer [11] in order to solve the resulting star set cover instance and obtain a disjoint collection of star sets that contain all nodes in V. We refer to star sets of size i as i-star sets. The algorithm first greedily includes disjoint 6-star sets in the solution until no other 6-star set can be included. We use the term maximal to refer to such collections of disjoint sets. After this phase, the nodes that remain uncovered and the star sets that consist only of such nodes form an instance of 5-star set cover. Then, the algorithm executes a restricted phase in order to pick a maximal collection of disjoint 5-star sets and guarantees that the number of 1-star sets in the final solution is not larger than the number of 1-star sets in the optimal solution of the 5-star set cover instance at the beginning of this restricted phase. Then, it applies a similar restricted phase for 4-star sets. After this phase, an instance of 3-star set cover remains to be solved in order to complete the star set covering; the algorithm applies a semi-local optimization phase on it. We use integers $6, 5, 4, 3$ to refer to the phases of the algorithm according to the size of the star sets they consider. In summary, the algorithm can be described as follows.

Phase 6: Choose a maximal collection of disjoint 6-star sets.
Phase 5: Choose a maximal collection of disjoint 5-star sets so that the choice of these star sets does not increase the number of 1-star sets in the final solution.
Phase 4: Choose a maximal collection of disjoint 4-star sets so that the choice of these star sets does not increase the number of 1-star sets in the final solution.
Phase 3: Run the semi-local optimization algorithm on the remaining instance of 3-star set cover.

The output of the algorithm is a disjoint set $\mathcal{T} \subseteq \mathcal{S}$ of star sets. We transform this solution to a spanning star forest as follows. We first consider star sets of $\mathcal{T}$ of type A. For each such set s, we set the common neighbor u of the nodes in s as center and connect the remaining nodes of s to u. Then, we consider the star sets of type B. For each such set s in $\mathcal{T}$, we select a common neighbor u of the nodes in s. If u is already the center of another star, we simply connect the nodes of s that have not been used as centers so far to it. If u is already the leaf of another star, we remove it from that star and connect the nodes of s that have not been used as centers so far to it.

Notice that, once a node has been used as a center, it remains one until the end of the algorithm and each leaf is connected to some center. Since the star sets include all nodes of V, each node is either a center or a leaf of some star and, hence, the solution is indeed a spanning star forest of G. Furthermore, when considering a star set, we increase the number of centers by at most 1. Hence, the total number of leaves is at least $|V| - |\mathcal{T}|$.

3 Analysis

Our main argument for the analysis of the algorithm can be described as follows. We first show that if there is an instance on which the algorithm computes an at most f-approximate solution, then an appropriately defined linear program LP(f) is feasible. Then, we show that LP(f) is infeasible for $f < 193/240$, implying that the approximation ratio of the algorithm is at least $193/240 \approx 0.804$. We consider only non-trivial instances of the problem in which the input graph has at least one edge since any algorithm is optimal for trivial ones.

3.1 The Parameterized LP Lemma

Consider a non-trivial SSF instance I that consists of a graph $G = (V, E)$ on which the algorithm computes an at most f-approximate solution. Let $(V, \mathcal{S})$ be the corresponding star set cover instance. For $i = 5, 4, 3$, denote by (V_i, S_i) the instance of the i-star set cover problem that has to be solved just before entering phase i. Here, V_i contains the nodes in V that have not been covered in previous phases and S_i contains the star sets of $\mathcal{S}$ which consist only of nodes in V_i. S_i contains star sets of the original collection of size at most i. Denote by $\mathcal{O}_i$ an optimal solution of instance (V_i, S_i); we also denote the optimal solution of $(V, \mathcal{S})$ by $\mathcal{O}$. Since $\mathcal{S}$ is closed under subsets, without loss of generality, we may assume that $\mathcal{O}_i$ contains disjoint sets. Furthermore, it is clear that $|\mathcal{O}_{i-1}| \leq |\mathcal{O}_i|$ for $i = 3, 4, 5$ and $|\mathcal{O}_i| \leq |\mathcal{O}|$.

Set $\theta = \frac{|V|}{|V|-1}$. We denote by T the ratio $|V|/|\mathcal{O}|$. Since the instance is non-trivial, there exists an optimal star cover with at most $|V| - 1$ star sets (i.e., $|\mathcal{O}| \leq |V| - 1$). Hence,

$$T \geq \theta. \tag{1}$$

For the phase i of the algorithm with $i = 3, 4, 5$, we denote by $a_{i,j}$ the ratio of the number of j-star sets in $\mathcal{O}_i$ over the number $|\mathcal{O}|$ of sets in the optimal solution of $(V, \mathcal{S})$. Since $|\mathcal{O}_i| \leq |\mathcal{O}|$ and $|\mathcal{O}_i| = |\mathcal{O}| \sum_{j=1}^{i} a_{i,j}$, we obtain that

$$\sum_{j=1}^{i} a_{i,j} \leq 1. \tag{2}$$

Also, observe that $|V| = T|\mathcal{O}|$ and $|V_5| = |\mathcal{O}| \sum_{j=1}^{5} j a_{5,j}$. Since $V_5 \subseteq V$, we have

$$T \geq \sum_{j=1}^{5} j a_{5,j}. \tag{3}$$

During the phase i with $i = 3, 4, 5$, the algorithm selects a maximal set of i-star sets. This means that any i-star set selected intersects some of the i-star sets of the optimal star set covering of the instance (V_i, S_i) and may intersect with at most i such i-star sets. Since there are $a_{i,i}|\mathcal{O}|$ such star sets in the optimal star set covering of (V_i, S_i), the number $|V_i \setminus V_{i-1}|/i$ of i-star sets selected during phase i is at least $a_{i,i}|\mathcal{O}|/i$. Equivalently, $|V_i \setminus V_{i-1}| \geq a_{i,i}|\mathcal{O}|$. Since $|V_i \setminus V_{i-1}| = \left(\sum_{j=1}^{i} j a_{i,j} - \sum_{j=1}^{i-1} j a_{i-1,j}\right) |\mathcal{O}|$, we obtain that

$$\sum_{j=1}^{i} j a_{i,j} - \sum_{j=1}^{i-1} j a_{i-1,j} \geq a_{i,i}. \tag{4}$$

Phase 3 imposes several extra constraints. Denote by b_3, b_2, b_1 the number of 3-, 2-, and 1-star sets computed by the semi-local optimization phase divided by $|\mathcal{O}|$. We use the main result of the analysis of Duh and Fürer [11] expressed in our notation.

Theorem 1 (Duh and Fürer[11]). *$b_1 \leq \alpha_{3,1}$ and $b_2 + b_1 \leq \alpha_{3,3} + \alpha_{3,2} + \alpha_{3,1}$.*

In addition, the restricted phase i (with $i = 4, 5$) imposes the constraint that the number of the 1-sets in the final solution does not increase compared to the number of 1-sets in the optimal star set covering of (V_i, S_i). Taking into account the first inequality of Theorem 1, we have

$$b_1 \leq a_{i,1}, \text{ for } i = 3, 4, 5. \tag{5}$$

Up to now, we have expressed all the properties of the instance $(V, \mathcal{S})$ as well as the behavior of the algorithm on it, besides the fact that the star set covering obtained implies an at most f-approximate solution for the corresponding SSF instance. We express the benefit of the algorithm in terms of our variables as follows. We denote by t_i the number of star sets computed during the phase i. For phase 6, we have

$$t_6 = \frac{1}{6}|V \setminus V_5| = \frac{1}{6}\left(T - \sum_{j=1}^{5} j a_{5,j}\right) |\mathcal{O}|. \tag{6}$$

For the phase i with $i = 4, 5$, we have

$$t_i = \frac{1}{i}|V_i \setminus V_{i-1}| = \frac{1}{i}\left(\sum_{j=1}^{i} j a_{i,j} - \sum_{j=1}^{i-1} j a_{i-1,j}\right)|\mathcal{O}|. \tag{7}$$

Also, for the semi-local optimization phase, we have

$$\begin{aligned} t_3 &= (b_3 + b_2 + b_1)|\mathcal{O}| = \left(\frac{b_1}{3} + \frac{b_2 + b_1}{3} + \frac{3b_3 + 2b_2 + b_1}{3}\right)|\mathcal{O}| \\ &\leq \left(\frac{b_1}{3} + \frac{a_{3,3} + a_{3,2} + a_{3,1}}{3} + \frac{3a_{3,3} + 2a_{3,2} + a_{3,1}}{3}\right)|\mathcal{O}| \\ &= \left(\frac{1}{3}b_1 + \frac{4}{3}a_{3,3} + a_{3,2} + \frac{2}{3}a_{3,1}\right)|\mathcal{O}|. \end{aligned} \tag{8}$$

The inequality follows by Theorem 1 and since $(3b_3 + 2b_2 + b_1)|\mathcal{O}| = (3a_{3,3} + 2a_{3,2} + a_{3,1})|\mathcal{O}| = |V_3|$.

By the discussion in Section 2, the solution obtained by the algorithm on the SSF instance I has benefit $ALG(I)$ at least $|V| - \sum_{i=3}^{6} t_i$. Consider an optimal spanning star forest of the graph G of instance I and let $OPT(I)$ be its benefit. This naturally corresponds to a star set cover $\mathcal{O}'$ for $(V, \mathcal{S})$ that consists of disjoint star sets of type A. Clearly, $OPT(I) = |V| - |\mathcal{O}'|$ and $|\mathcal{O}| \leq |\mathcal{O}'|$. Hence, $OPT(I) \leq |V| - |\mathcal{O}|$. Since the solution obtained by the algorithm is at most f-approximate (i.e., $ALG(I) \leq f \cdot OPT(I)$), we have $|V| - \sum_{i=3}^{6} t_i \leq f\,(|V| - |\mathcal{O}|)$ and, equivalently,

$$(f-1)|V| + \sum_{i=3}^{6} t_i \geq f|\mathcal{O}|. \tag{9}$$

We use (6), (7), and (8) to upper-bound the left side of inequality (9). We have

$$\begin{aligned} &(f-1)|V| + \sum_{i=3}^{6} t_i \\ &\leq (f-1)T|\mathcal{O}| + \frac{1}{6}\left(T - \sum_{j=1}^{5} j a_{5,j}\right)|\mathcal{O}| + \frac{1}{5}\left(\sum_{j=1}^{5} j a_{5,j} - \sum_{j=1}^{4} j a_{4,j}\right)|\mathcal{O}| \\ &\quad + \frac{1}{4}\left(\sum_{j=1}^{4} j a_{4,j} - \sum_{j=1}^{3} j a_{3,j}\right)|\mathcal{O}| + \left(\frac{1}{3}b_1 + \frac{4}{3}a_{3,3} + a_{3,2} + \frac{2}{3}a_{3,1}\right)|\mathcal{O}| \\ &= \left(\left(f - \frac{5}{6}\right)T + \frac{1}{6}a_{5,5} + \frac{2}{15}a_{5,4} + \frac{1}{10}a_{5,3} + \frac{1}{15}a_{5,2} + \frac{1}{30}a_{5,1} + \frac{1}{5}a_{4,4}\right. \\ &\quad \left. + \frac{3}{20}a_{4,3} + \frac{1}{10}a_{4,2} + \frac{1}{20}a_{4,1} + \frac{7}{12}a_{3,3} + \frac{1}{2}a_{3,2} + \frac{5}{12}a_{3,1} + \frac{1}{3}b_1\right)|\mathcal{O}|. \end{aligned} \tag{10}$$

By (9) and (10), we obtain

$$(f-\frac{5}{6})T+\frac{1}{6}a_{5,5}+\frac{2}{15}a_{5,4}+\frac{1}{10}a_{5,3}+\frac{1}{15}a_{5,2}+\frac{1}{30}a_{5,1}+\frac{1}{5}a_{4,4}$$
$$+\frac{3}{20}a_{4,3}+\frac{1}{10}a_{4,2}+\frac{1}{20}a_{4,1}+\frac{7}{12}a_{3,3}+\frac{1}{2}a_{3,2}+\frac{5}{12}a_{3,1}+\frac{1}{3}b_1 \geq f. \quad (11)$$

By expressing inequalities (1)-(5) and (11) in standard form, we obtain our parameterized LP lemma.

Lemma 1. *If there exists an instance I of* SSF *for which the algorithm computes a solution of benefit $ALG(I) \leq f \cdot OPT(I)$ for some $f \in [0,1]$, then the following linear program $LP(f)$ has a feasible solution for some $\theta > 1$.*

$$T \geq \theta$$
$$-\sum_{j=1}^{i} a_{i,j} \geq -1, \quad \textit{for } i = 3,4,5$$
$$T - \sum_{j=1}^{5} j a_{5,j} \geq 0$$
$$(i-1)a_{i,i} + \sum_{j=1}^{i-1} j a_{i,j} - \sum_{j=1}^{i-1} j a_{i-1,j} \geq 0, \quad \textit{for } i = 4,5$$
$$a_{i,1} - b_1 \geq 0, \quad \textit{for } i = 3,4,5$$
$$(f-\frac{5}{6})T+\frac{1}{6}a_{5,5}+\frac{2}{15}a_{5,4}+\frac{1}{10}a_{5,3}+\frac{1}{15}a_{5,2}+\frac{1}{30}a_{5,1}+\frac{1}{5}a_{4,4}$$
$$+\frac{3}{20}a_{4,3}+\frac{1}{10}a_{4,2}+\frac{1}{20}a_{4,1}+\frac{7}{12}a_{3,3}+\frac{1}{2}a_{3,2}+\frac{5}{12}a_{3,1}+\frac{1}{3}b_1 \geq f$$
$$a_{i,j} \geq 0, \quad \textit{for } i = 3,4,5 \textit{ and } j = 1,...,i$$
$$b_1 \geq 0$$

3.2 Proof of the Approximation Bound

The proof of the approximation bound is based on the following lemma.

Lemma 2. *For every $f < 193/240$, $LP(f)$ has no feasible solution.*

Proof. We can assume that LP(f) is a minimization linear program with objective 0. By duality, if it were feasible, then the optimal objective value of the dual maximization linear program should be 0 as well. We show that this is not the case and that the dual has a solution with strictly positive objective value. This implies the lemma.

In the dual LP, we use the eleven variables η, β_3, β_4, β_5, γ_4, γ_5, γ_6, δ_3, δ_4, δ_5 and ζ corresponding to the constraints of LP(f). Variable η corresponds to the first constraint of LP(f), β_i correspond to the second set of constraints, γ_6 corresponds to the third constraint, γ_4 and γ_5 correspond to the fourth set of

constraints, δ_i correspond to the fifth set of constraints, and ζ corresponds to the last constraint. So, the dual of LP(f) is

$$\begin{aligned}
\text{maximize} \quad & \theta\eta - \beta_3 - \beta_4 - \beta_5 + f\zeta \\
\text{subject to} \quad & \eta + \gamma_6 + (f - 5/6)\zeta \leq 0 \\
& -\delta_3 - \delta_4 - \delta_5 + \zeta/3 \leq 0 \\
& -\beta_3 - \gamma_4 + \delta_3 + 5\zeta/12 \leq 0 \\
& -\beta_3 - 2\gamma_4 + \zeta/2 \leq 0 \\
& -\beta_3 - 3\gamma_4 + 7\zeta/12 \leq 0 \\
& -\beta_4 + \gamma_4 - \gamma_5 + \delta_4 + \zeta/20 \leq 0 \\
& -\beta_4 + 2\gamma_4 - 2\gamma_5 + \zeta/10 \leq 0 \\
& -\beta_4 + 3\gamma_4 - 3\gamma_5 + 3\zeta/20 \leq 0 \\
& -\beta_4 + 3\gamma_4 - 4\gamma_5 + \zeta/5 \leq 0 \\
& -\beta_5 + \gamma_5 - \gamma_6 + \delta_5 + \zeta/30 \leq 0 \\
& -\beta_5 + 2\gamma_5 - 2\gamma_6 + \zeta/15 \leq 0 \\
& -\beta_5 + 3\gamma_5 - 3\gamma_6 + \zeta/10 \leq 0 \\
& -\beta_5 + 4\gamma_5 - 4\gamma_6 + 2\zeta/15 \leq 0 \\
& -\beta_5 + 4\gamma_5 - 5\gamma_6 + \zeta/6 \leq 0 \\
& \beta_i, \delta_i \geq 0, \quad \text{for } i = 3, 4, 5 \\
& \gamma_i \geq 0, \quad \text{for } i = 4, 5, 6 \\
& \zeta, \eta \geq 0
\end{aligned}$$

The solution $\eta = 193/240 - f$, $\beta_3 = 39/72$, $\beta_4 = 11/72$, $\beta_5 = 79/720$, $\gamma_4 = 1/72$, $\gamma_5 = 1/45$, $\gamma_6 = 7/240$, $\delta_3 = 5/36$, $\delta_4 = 1/9$, $\delta_5 = 1/12$, and $\zeta = 1$ satisfies all the constraints. Observe that $\eta - \beta_3 - \beta_4 - \beta_5 + f\zeta = 0$ and, hence, the objective value is $(\theta - 1)\eta = (\theta - 1)(193/240 - f) > 0$. The lemma follows. □

Theorem 2. *The approximation ratio of the algorithm is at least* 193/240.

Proof. By Lemmas 1 and 2, we have that for any $f < 193/240$ and for any instance I of SSF, the algorithm computes a solution of benefit $ALG(I) > f \cdot OPT(I)$. Hence, its approximation ratio is at least 193/240. □

3.3 A Note for Color Saving

We point out that neither the algorithm nor the analysis makes use of the fact that the star sets actually correspond to stars in the graph. Hence, the algorithm applies to the more general complementary set cover problem where the sets are given explicitly. Also, since the algorithm considers sets of constant size, it also applies to color saving. What is required is to replace the term "star set" by the term "independent set". Although computing an independent set of a given size

on a graph is a classical NP-hard problem, computing independent sets of size up to 6 as the algorithm requires can be done easily in polynomial time. Our analysis is tight; the lower bound construction is omitted.

4 A Lower Bound for Natural Local Search SSF Algorithms

In this section we consider a natural family of local search algorithms for SSF. We consider solutions of instances of SSF as assignments of values 0 and 1 to the nodes of the input graph where 0 or 1 at a node means that the node is a center or leaf, respectively. Such an assignment corresponds to a feasible solution of SSF when each node that has been assigned value 1 is adjacent to at least one node that has been assigned value 0. The benefit of a feasible assignment is then the number of nodes that have been assigned value 1.

Local search can be used as follows. Starting with any feasible assignment (e.g., with 0 to each node), a local search algorithm repeatedly performs k-changes (for some constant integer k) while this is possible. Performing a k-change means to alter the assignment of $t \leq k$ nodes originally having value 1 to 0 and the assignment of $t+1$ nodes originally assigned value 0 to 1. The algorithm terminates when a local optimum assignment is reached, i.e., one from which no k-change is possible. Note that since k is constant and the benefit increases by 1 at each step, the algorithm terminates in polynomial time. The algorithm is efficient if all local optima have high benefit.

The local search algorithm that performs 0-changes may have very poor approximation ratio. Indeed, consider the instance that consists of two nodes u and v that are assigned the value 1 and $n-2$ nodes u_1, ..., u_{n-2} that are connected to both u and v and are all assigned the value 0. This assignment has a benefit of 2 and is a local optimum since no 0-change is possible, while the solution that assigns 1 to nodes u_1, ..., u_{n-2} and 0 to nodes u and v is feasible and has a benefit of $n-2$. Furthermore, we can show that the local search algorithm that performs 1-changes always computes an 1/2-approximate assignment. This could indicate that better bounds can be obtained by considering k-changes with higher constant values of k. Unfortunately, this is not the case as the following theorem states.

Theorem 3. *For every integer $k \geq 0$ and every $\epsilon \in \left(0, \frac{1}{2(k+2)}\right]$, the local-search algorithm that performs $k-$changes has an $(1/2+\epsilon)$-approximate solution as a local optimum.*

Proof. The proof uses a result of Erdös and Sachs [12] stating that, for every integer $d, g > 0$, there exists a d-regular graph of girth at least g.

Our starting point is a d-regular graph G of girth at least g, where $d = 2\lceil 1/2\epsilon \rceil$ and $g = k+2$. Given G, consider the graph G' that is obtained by replacing each edge (u, v) of G by a path of size 3 (henceforth called a 3-path) $\langle u, Z^1_{uv}, Z^2_{uv}, v\rangle$. So, the graph G has two additional nodes Z^1_{uv} and Z^2_{uv} for each edge (u, v) in G. We refer to these nodes as edge-nodes.

We define an assignment on the nodes of G' which has (asymptotically) half the optimal benefit and which cannot be improved by a k-change. In order to construct this assignment, we make use of the fact that a connected graph has an Euler circuit when all its nodes have even degree. We consider such a directed Euler circuit in G' and assign value 1 to the first node of each 3-path the Euler circuit comes across and value 0 to the other node of the 3-path. Furthermore, we assign value 1 to every non-edge-node. We notice that this is a feasible assignment since every edge-node with value 1 is connected to the other edge-node of the same 3-path having value 0, and every non-edge-node has exactly $d/2 = \lceil 1/2\epsilon \rceil \geq k + 2$ neighbors with value 0.

We use proof by contradiction. Suppose the aforementioned bad assignment is not a local optimum and consider another neighboring feasible assignment (that is obtained by applying a $k-$change to the original one). In more detail, assume that the improved assignment is obtained if we assign value 1 to $t+1$ edge-nodes having value 0 originally, as well as change the value of $s \leq t$ edge-nodes and of $t - s$ non-edge-nodes from 1 to 0.

Consider the intermediate assignment, where only the changes on the values of the edge-nodes have been applied. We notice that the constraints regarding the non-edge-nodes are not violated since each such node originally had at least $k + 2$ neighbors being assigned value 0 and at most $k + 1$ changes from 0 to 1 have been applied. Regarding the edge-nodes, the only ones whose constraint is unsatisfied in the intermediate assignment belong to 3-paths in which both edge-nodes have value 1. The number of such 3-paths is at least $t + 1 - s$ and at most $k + 1$. We consider the edge-induced graph H consisting of the edges of G that correspond to these 3-paths. Since H has at most $k+1$ edges and the girth of G is at least $k+2$, H is a forest. In addition, the nodes of graph H correspond to the $t - s$ non-edge-nodes of G' whose value is 0 at the final assignment. As we have already mentioned, graph H has at least $t + 1 - s$ edges, which implies that it contains a cycle. Furthermore H has at most $k + 1$ edges, therefore its girth is at most $k + 1$, which contradicts the original assumption.

We conclude that no $k-$change can be applied to the bad assignment above and it is a local optimum with benefit $n(1+d/2)$ while the benefit of the optimal assignment on G' is dn, which is achieved when only non-edge-nodes are assigned value 0. Hence, the approximation ratio is at most $1/2 + 1/d \leq 1/2 + \epsilon$. □

References

1. Agra, A., Cardoso, D., Cerfeira, O., Rocha, E.: A spanning star forest model for the diversity problem in automobile industry. In: Proceedings of ECCO XVII (2005)
2. Athanassopoulos, S., Caragiannis, I., Kaklamanis, C.: Analysis of approximation algorithms for k-set cover using factor-revealing linear programs. In: Csuhaj-Varjú, E., Ésik, Z. (eds.) FCT 2007. LNCS, vol. 4639, pp. 52–63. Springer, Heidelberg (2007)
3. Ausiello, G., D'Atri, A., Protasi, M.: Structure preserving reductions among convex optimization problems. Journal of Computer and System Sciences 21, 136–153 (1980)
4. Ausiello, G., Marchetti-Spaccamela, A., Protasi, M.: Toward a unified approach for the classification of NP-complete optimation problems. Theoretical Computer Science 12, 83–96 (1980)

5. Berry, V., Guillemot, S., Nicholas, F., Paul, C.: On the approximation of computing evolutionary trees. In: Wang, L. (ed.) COCOON 2005. LNCS, vol. 3595, pp. 115–125. Springer, Heidelberg (2005)
6. Caragiannis, I.: Wavelength management in WDM rings to maximize the number of connections. SIAM Journal on Discrete Mathematics 23(2), 959–978 (2009)
7. Chakrabarty, D., Goel, G.: On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. In: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008), pp. 687–696 (2008)
8. Chen, N., Engelberg, R., Nguyen, C.T., Raghavendra, P., Rudra, A., Singh, G.: Improved approximation algorithms for the spanning star forest problem. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) RANDOM 2007 and APPROX 2007. LNCS, vol. 4627, pp. 44–58. Springer, Heidelberg (2007)
9. Demange, M., Paschos, V.T.: On an approximation measure founded on the links between optimization and polynomial approximation theory. Theoretical Computer Science 158, 117–141 (1996)
10. Demange, M., Grisoni, P., Paschos, V.T.: Differential approximation algorithms for some combinatorial optimization problems. Theoretical Computer Science 209, 107–122 (1998)
11. Duh, R., Fürer, M.: Approximation of k-set cover by semi local optimization. In: Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC 1997), pp. 256–264 (1997)
12. Erdös, P., Sachs, H.: Reguläre Graphen gegebener Taillenweite mit minimaler Knotenzahl. Wiss. Z. Univ. Halle, Math.-Nat. 12, 251–258 (1963)
13. Feige, U.: A threshold of $\ln n$ for approximating set cover. Journal of the ACM 45(4), 634–652 (1998)
14. Halldórsson, M.M.: Approximating discrete collections via local improvements. In: Proceedings of the 6th Annual ACM/SIAM Symposium on Discrete Algorithms (SODA 1995), pp. 160–169 (1995)
15. Halldórsson, M.M.: Approximating k-set cover and complementary graph coloring. In: Cunningham, W.H., Queyranne, M., McCormick, S.T. (eds.) IPCO 1996. LNCS, vol. 1084, pp. 118–131. Springer, Heidelberg (1996)
16. Hassin, R., Khuller, S.: z-Approximations. Journal of Algorithms 41(2), 429–442 (2001)
17. Hurkens, C.A.J., Schrijver, A.: On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. SIAM Journal on Discrete Mathematics 2(1), 68–72 (1989)
18. Jain, K., Mahdian, M., Markakis, E., Saberi, A., Vazirani, V.V.: Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. Journal of the ACM 50(6), 795–824 (2003)
19. Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences 9, 256–278 (1974)
20. Levin, A.: Approximating the unweighted k-set cover problem: greedy meets local search. SIAM Journal on Discrete Mathematics 23(1), 251–264 (2009)
21. Lovász, L.: On the ratio of optimal integral and fractional covers. Discrete Mathematics 13, 383–390 (1975)
22. Nguyen, C.T., Shen, J., Hou, M., Sheng, L., Miller, W., Zhang, L.: Approximating the spanning star forest problem and its application to genomic sequence alignment. SIAM Journal on Computing 38(3), 946–962 (2008)