

# Everywhere-Sparse Spanners via Dense Subgraphs

Eden Chlamtác\*  
Tel Aviv University

Michael Dinitz†  
The Weizmann Institute

Robert Krauthgamer†  
The Weizmann Institute

June 7, 2018

## Abstract

The significant progress in constructing graph spanners that are sparse (small number of edges) or light (low total weight) has skipped spanners that are everywhere-sparse (small maximum degree). This disparity is in line with other network design problems, where the maximum-degree objective has been a notorious technical challenge. Our main result is for the Lowest Degree 2-Spanner (LD2S) problem, where the goal is to compute a 2-spanner of an input graph so as to minimize the maximum degree. We design a polynomial-time algorithm achieving approximation factor  $\tilde{O}(\Delta^{3-2\sqrt{2}}) \approx \tilde{O}(\Delta^{0.172})$ , where  $\Delta$  is the maximum degree of the input graph. The previous  $\tilde{O}(\Delta^{1/4})$ -approximation was proved nearly two decades ago by Kortsarz and Peleg [SODA 1994, SICOMP 1998].

Our main conceptual contribution is to establish a formal connection between LD2S and a variant of the Densest k-Subgraph (DkS) problem. Specifically, we design for both problems strong relaxations based on the Sherali-Adams linear programming (LP) hierarchy, and show that “faithful” randomized rounding of the DkS-variant can be used to round LD2S solutions. Our notion of faithfulness intuitively means that all vertices and edges are chosen with probability proportional to their LP value, but the precise formulation is more subtle.

Unfortunately, the best algorithms known for DkS use the Lovász-Schrijver LP hierarchy in a non-faithful way [Bhaskara, Charikar, Chlamtac, Feige, and Vijayaraghavan, STOC 2010]. Our main technical contribution is to overcome this shortcoming, while still matching the gap that arises in random graphs by planting a subgraph with same log-density.

## 1 Introduction

The significant progress made over the years in constructing graph spanners shares, for the most part, two features: (1) the objective is to minimize the total number/weight of edges; and (2) the techniques are primarily combinatorial. This second feature has started to change recently, with the use of Linear Programming (LP) in several results [BGJ<sup>+</sup>09, BRR10, DK11a, BBM<sup>+</sup>11]. One of the earliest uses of linear programming for spanners, though, was also one of the few examples of a different objective function: in 1994, Kortsarz and Peleg [KP98] considered the LOWEST DEGREE 2-SPANNER (LD2S) problem, where the goal is to find a 2-spanner of an input graph that minimizes the maximum degree, and used a natural LP relaxation to devise a polynomial-time algorithm achieving approximation factor  $\tilde{O}(\Delta^{1/4})$  (where  $\Delta$  is the maximum degree). They also showed

---

\*Research supported in part by an ERC Advanced grant. Email: `chlamtac@post.tau.ac.il`

†Work supported in part by an Israel Science Foundation grant #452/08, a US-Israel BSF grant #2010418, and by a Minerva grant. Email: `{michael.dinitz,robert.krauthgamer}@weizmann.ac.il`

that it is NP-hard to approximate LD2S within a factor smaller than  $\Omega(\log n)$ . We make the first progress on approximating LD2S since then, by designing a new approximation algorithm with an improved approximation factor.

**Theorem 1.1.** *For an arbitrarily small fixed  $\varepsilon > 0$ , the LD2S problem can be approximated in polynomial time within factor  $\tilde{O}(\Delta^{3-2\sqrt{2}+\varepsilon}) \leq \tilde{O}(\Delta^{0.172})$ .*

Degree bounds have a natural mathematical appeal and are also useful in many applications. For example, one common use of spanners is in compact routing schemes (e.g. [TZ01, Din07]), which store small routing tables at every node. If we route on a spanner with large maximum degree, then *a priori* the node of large degree will have a large table, even if the total number of edges is small. Similarly, the maximum degree (rather than the overall number of edges) is what determines local memory constraints when using spanners to construct network synchronizers [PU89] or for efficient broadcast [ABP92]. The literature on approximation algorithms includes recent exciting work on sophisticated LP rounding for network design problems involving degree bounds (e.g. [LNSS09, SL07]).

**Dense subgraphs.** Our central insight involves the relationship between *sparse* spanners and finding *dense* subgraphs. Such an informal relationship has been folklore in the distributed computing and approximation algorithms communities; for instance, graph spanners are mentioned as the original motivation for introducing the DENSEST  $k$ -SUBGRAPH (DkS) problem [KP93], in that case in the context of minimizing the total number of edges in the spanner. Surprisingly, we show that there is a natural connection between DkS and the more challenging task of constructing spanners that have small maximum degree. We prove that certain types of “faithful” approximation algorithms for a variant of DkS which we call SMALLEST  $m$ -EDGE SUBGRAPH (or smES) imply approximation algorithms for LD2S, and then show how to construct such an algorithm for smES; combining these two together yields our improved approximation for LD2S.

We seem to be the first to formally define and study smES, although it has been used in previous work (sometimes implicitly) as the natural minimization version of DkS, see e.g. [Nut10, GHNR07, AGGN10]. A straightforward argument shows that an  $f$ -approximation for smES implies an  $\tilde{O}(f^2)$ -approximation for DkS. In the other direction, all that was known was that an  $f$ -approximation for DkS implies an  $\tilde{O}(f)$ -approximation for smES. One contribution of this paper is a non-black box improvement: while the best-known approximation for DkS is  $O(n^{1/4+\varepsilon})$ , we give an  $O(n^{3-2\sqrt{2}+\varepsilon})$ -approximation for smES. This improvement is key to our main result about approximating LD2S.

**LP hierarchies.** The log-density framework introduced in [BCC<sup>+</sup>10] in the context of DkS (see Section 2.2.1) predicts, when applied to smES, that current techniques would hit a barrier at  $n^{3-2\sqrt{2}}$ , precisely the factor achieved by our algorithm. Here, the use of strong relaxations (namely LP hierarchies) is crucial, since simple relaxations have large integrality gaps. For example, one can show that the natural SDP relaxation for smES has an  $\Omega(n^{1/4})$  integrality gap (for  $G = G(n, n^{-1/2})$  and  $m = n^{1/2}$ ), similarly to the  $\Omega(n^{1/3})$ -gap shown for DkS by Feige and Seltser [FS97].

While we borrow some of the algorithmic techniques developed for DkS by [BCC<sup>+</sup>10], the crucial need for a “faithful” approximation required us to develop new tools which represent a significant departure from previous work both in terms of the algorithm and its analysis. For example, our algorithm and analysis rely on the existence of consistent high-moment variables arising from the

Sherali-Adams [SA90] hierarchy (see, e.g. Lemma 6.3) and not present in the Lovász-Schrijver [LS91] LP hierarchy (which was sufficient for [BCC<sup>+</sup>10]).

**Basic terminology.** We denote the (undirected)<sup>1</sup> input graph by  $G = (V, E)$ , and let  $n = |V|$ . For a vertex  $v \in V$ , let  $\Gamma_G(v) = \{u : \{u, v\} \in E\}$  denote its neighbors in  $G$ . If the graph  $G$  is clear from context then we will drop the subscript and simply refer to  $\Gamma(v)$ . Recall that the maximum degree of vertices in  $G$  is denoted  $\Delta$ . We suppress polylogarithmic factors by using the notation  $\tilde{O}(f)$  as a shorthand for  $f \cdot (\log n)^{O(1)}$ .

As usual, a *2-spanner* of  $G$  is a subgraph  $H = (V, E_H)$  such that every  $u, v \in V$  that are connected by an edge in  $G$  are also connected in  $H$  by a path of length at most 2. This is a special case of the more general notion of a *k-spanner*, which was introduced by Peleg and Schäffer [PS89] and has been studied extensively; see also Section 1.4.

### 1.1 LP-based approach for LD2S

The LP relaxation of LD2S used by Kortsarz and Peleg [KP98] is very natural: for each edge  $\{u, v\} \in E$  it has a variable  $x_{\{u, v\}} \in [0, 1]$ , plus additional variables  $x_{\{u, v\}; w} \in [0, 1]$  for every  $w \in \Gamma(u) \cap \Gamma(v)$  (i.e., whenever  $u, v, w$  form a triangle in  $G$ ). The objective is to minimize  $\lambda$ , subject to a degree constraint

$$\sum_{v \in \Gamma(u)} x_{\{u, v\}} \leq \lambda \quad \forall u \in V, \tag{1}$$

and the constraints that every edge in  $G$  (i.e. demand pair) is covered by either a 1-path or a 2-path in the spanner (subgraph):

$$x_{\{u, v\}} + \sum_{w \in \Gamma(u) \cap \Gamma(v)} x_{\{u, v\}; w} \geq 1 \quad \forall \{u, v\} \in E. \tag{2}$$

$$x_{\{u, v\}; w} \leq \min\{x_{\{u, w\}}, x_{\{v, w\}}\} \quad \forall \{u, v\} \in E, w \in \Gamma(u) \cap \Gamma(v). \tag{3}$$

This LP relaxation seems like a natural place to start, but it is actually quite weak, having integrality gap  $\Omega(\sqrt{\Delta})$ . Indeed, let  $G$  be a clique of size  $\Delta + 1$ ; observe that every 2-spanner of this  $G$  must have maximum degree at least  $\sqrt{\Delta}$ , while the LP has value  $\lambda \leq 1$  (by setting all  $x$  variables to  $1/\Delta$ ). The same argument works for a disjoint union of  $n/(\Delta + 1)$  such cliques. Kortsarz and Peleg [KP98] nevertheless managed to achieve  $\tilde{O}(\Delta^{1/4})$  approximation (in polynomial-time). Their algorithm combines a relatively simple rounding of this LP with another partial solution that does not use the LP, and whose analysis relies on a combinatorial lower bound on the optimum.

Our approach is to look at the Kortsarz-Peleg LP above from the perspective of a single vertex  $w$ . Consider an integral solution  $H$  to the LP above, i.e. a valid 2-spanner. From the viewpoint of  $w$ , incident edges are included in  $H$  for two possible reasons: either to span an edge connecting two neighbors of  $w$  (i.e., including the edges  $\{u, w\}$  and  $\{v, w\}$  in order to span the edge  $\{u, v\}$ ), or to span the edge itself. It's reasonable to focus on the case where  $H$  has significantly fewer edges than  $G$ , and therefore many edges in  $H$  are included because of the first reason. Let  $G_w$  be the subgraph of  $G$  induced by the neighbors of  $w$ , and let  $S$  be the subset of vertices of  $G_w$  that are adjacent to  $w$  in  $H$ . Then from the perspective of  $w$ , including the edges between  $w$  and  $S$  in  $H$  “covers” every demand formed by an edge (of  $G_w$ ) that connects two vertices in  $S$ , namely

---

<sup>1</sup>Our algorithm for LD2S also works for the directed case, though for simplicity we focus on undirected graphs.

$E' = \{\{u, v\} \in E : u, v \in S\}$ . We can look at each neighborhood this way, and reinterpret LD2S as the problem of covering every demand in at least one neighborhood  $G_w$ , while minimizing the maximum degree.

This viewpoint naturally suggests an LP-based algorithm for LD2S: solve the Kortsarz-Peleg LP above (or some other relaxation), and for every  $w \in V$ , interpret  $\sum_{\{u,v\} \in E: u,v \in \Gamma(w)} x_{\{u,v\};w}$  as the amount of “demand” that  $w$  is supposed to cover locally, and  $\sum_{u \in \Gamma(w)} x_{\{w,u\}}$  as  $w$ ’s “budget”. Then for each  $w \in V$  run a subroutine that covers the required amount of demand within the budget. Since in LD2S every demand *must* be covered, this subroutine should cover the required amount of demand but is free to somewhat violate the budget constraint; the amount of violation will correspond to the LD2S approximation guarantee. We thus need to solve the SMALLEST  $m$ -EDGE SUBGRAPH (*smES*) problem: given a graph (in our case  $G_w$ ) and a value  $m$ , choose as few vertices as possible subject to covering at least  $m$  edges, where an edge is covered if both its endpoints are chosen.

Unfortunately, this reduction from LD2S to *smES* does not work. There are two main issues with it. First, if  $w$  chooses to add an edge to  $u$  (i.e. the *smES* algorithm at  $G_w$  includes  $u \in \Gamma(w)$ ) then this increases the degree of *both*  $w$  and  $u$ . So even if  $u$  stays within its own budget when  $G_u$  is processed, many of its neighbors might decide to add their edge to  $u$ , and the degree at  $u$  will be very large compared to its budget. Second, since we run a *smES* algorithm at each vertex *separately*, they might make poorly-correlated choices as to which demands they cover. This may cause a high degree of overlap in the demands covered by different vertices, leading to much less total demand covered. Both of these problems stem from the same source: while we used the LP to define the total demand and budget at each vertex, we did not require the *smES* algorithm to act in a way consistent with the LP. If we could force the *smES* subroutine to make decisions that actually correspond to the fractional solution, then both of these problems would be solved. This is our motivation for defining faithfulness.

## 1.2 Faithful rounding

While our formal notion of faithfulness is somewhat technical and depends on the exact problem that we want to solve, the intuition behind it is natural and can apply to many problems. Suppose that we have an LP in which there are variables  $\{x_e\}_{e \in U}$  (where  $U$  is a universe of elements) as well as variables  $\{x_{e,e'}\}_{e,e' \in U}$ . In our case, each  $e$  is a vertex in a *smES* instance (i.e. an edge in LD2S) and each pair  $\{e, e'\}$  is an edge in a *smES* instance (a 2-path in LD2S). A standard way of interpreting fractional LP values is as *probabilities*, i.e. we think of  $x_e$  as the probability that  $e$  should be in the solution. This interpretation naturally leads to independent randomized rounding, where we take  $e$  into our solution with probability proportional to  $x_e$ . By this interpretation,  $x_{e,e'}$  should be the probability that both  $e$  and  $e'$  are in the solution. But now we have a problem, since the natural constraints to force this type of situation in an integral setting, namely constraints such as  $x_{e,e'} \leq \min\{x_e, x_{e'}\}$ , correspond poorly to the probabilities obtained by independent randomized rounding. For example, if  $x_e = x_{e'} = x_{e,e'}$ , then the LP “believes” that the probability that both  $e$  and  $e'$  are in the solution is  $x_{e,e'}$ , but under independent randomized rounding this event happens with probability  $x_e \cdot x_{e'} = x_{e,e'}^2$ , which could be much smaller. In a faithful rounding this does not happen: roughly speaking, faithfulness requires every element and pair of elements to be included in the solution with probability that is proportional to its LP value.

Many algorithms are naturally faithful, and indeed we suspect that one reason this notion has not been defined previously (to the best of our knowledge) is that in most cases it either

falls out from the analysis “for free” or it is unnecessary. The connection we show between LD2S and faithful rounding for  $smES$  might give one hope that the recent algorithmic breakthrough for  $DkS$  by Bhaskara, Charikar, Chlamtac, Feige and Vijayaraghavan [BCC<sup>+</sup>10] could imply better approximations for LD2S. However, their result heavily uses hierarchies, which creates a formidable obstacle for faithful rounding, as we discuss in Section 1.3.

### 1.3 LP hierarchies and faithful rounding

Following the lead of Bhaskara et al. [BCC<sup>+</sup>10], we employ a strong LP relaxation for  $smES$ , which can be viewed as part of an LP hierarchy. In this context, a hierarchy is a sequence of increasingly tight relaxations to a 0-1 program, usually obtained via a general mechanism that works for any 0-1 program. Such hierarchies (for both LPs and SDPs) have been suggested by Sherali and Adams [SA90], Lovász and Schrijver [LS91], and Lasserre [Las02] (in our case, we use the Sherali-Adams hierarchy). A key property shared by these hierarchies is that they are locally integral; that is, the  $q$ -th relaxation in the hierarchy coincides exactly with the convex hull of feasible 0-1 solutions, when both are projected onto any  $q$ -dimensional subspace corresponding to  $q$  variables in the program.<sup>2</sup> Specifically for Sherali-Adams, the  $q$ -th relaxation for a given 0-1 linear program with variables  $x_1, \dots, x_N \in \{0, 1\}$ , is obtained by extending the 0-1 program to include a variable  $x_S$  for every  $S \subseteq \{1, \dots, N\}$ ,  $|S| \leq q$ , and then writing a “locally integral” relaxation for this extended 0-1 program to guarantee that  $x_S = \prod_{i \in S} x_i$  (by convention  $x_\emptyset = 1$ ). For more details, see the survey [CT12].

There has been a recent surge of interest in the study of hierarchies of LPs (or other convex programs), especially in connection with approximation algorithms for combinatorial optimization problems. Specifically, such strong relaxations can potentially lead to progress on problems whose approximability has persistent gaps, such as VERTEX-COVER and MINIMUM-BISECTION. This line of attack was probably first described explicitly in [ABL02]. However, designing rounding procedures for these relaxations is often quite challenging. Indeed, relatively few papers have managed to improve over state-of-the-art approximation algorithms using hierarchies. The few papers that do give improved approximation bounds using hierarchies include [Chl07, CS08, BCG09, CKR10, BCC<sup>+</sup>10].<sup>3</sup> In particular, the last paper designs a rounding procedure for an LP hierarchy for  $DkS$ , which we adapt for  $smES$ .

Our plan is to leverage the success of [BCC<sup>+</sup>10], but as mentioned before, we face a serious obstacle — their rounding procedure is not faithful. They essentially condition on a small set of events, for instance that the solution includes a small set  $S^*$  of *carefully chosen* elements, and then they use only the LP variables for sets containing this  $S^*$ , namely, a variable  $x_{S^* \cup \{u\}}$  is now thought of as the LP variable for singleton  $u$ . But clearly that variable might have very little to do with the actual  $x_u$ , which is the quantity with respect to which we are trying to be faithful.

Our main technical contribution is to overcome this and design a faithful rounding for  $smES$  based on Sherali-Adams. Our algorithm is loosely based on the  $DkS$  algorithm of [BCC<sup>+</sup>10], but numerous technical difficulties have to be resolved to make it faithful. This, together with our reduction from LD2S to faithful  $smES$ , gives our new approximation algorithm for LD2S. We believe

---

<sup>2</sup>Consequently, if  $N$  denotes the number of initial 0-1 variables, then the  $N$ -th relaxation is exactly the convex hull of all 0-1 solutions, i.e., corresponds to solving the 0-1 optimization problem exactly. The  $q$ -th relaxation in the sequence can be written explicitly as a (linear) program of size  $N^{O(q)}$ , and thus solved in time  $N^{O(q)}$ .

<sup>3</sup>There are also papers that recover known approximation bounds, say a PTAS, while other ones show the limitations of these hierarchies by exhibiting integrality gaps for certain problems and hierarchies.

that our notion of faithful rounding is of independent interest, and might prove useful for other approximation algorithms, especially in the context of using hierarchies such as Sherali-Adams.

For comparison, we mention that recent algorithmic results, due to [BRS11, GS11], design rounding schemes for the Lasserre [Las02] hierarchy. Their rounding appears to be faithful (at least at an informal level), but it is not applicable to our context. First, their analysis holds only for expander-like graphs, and second, their rounding technique applies to problems such as constraint satisfaction and graph partitioning, with no connection to  $DkS$ .

## 1.4 Related work

Graph spanners, first introduced by Peleg and Schäffer [PS89] and Peleg and Ullman [PU89], have been studied extensively, with applications ranging from routing in networks (e.g. [AP95, TZ05]) to solving linear systems (e.g. [ST04, EEST08]). The foundational result on spanners is due to Althöfer, Das, Dobkin, Joseph and Soares [ADD<sup>+</sup>93], who gave an algorithm that, given a graph and an integer  $k \geq 1$ , constructs a  $(2k - 1)$ -spanner with  $n^{1+1/k}$  edges. Unfortunately this result obviously does not give anything nontrivial for 2-spanners, and indeed it is easy to see that there exist graphs for which every 2-spanner has  $\Omega(n^2)$  edges, thus nontrivial absolute bounds on the size of a 2-spanner are not possible. Kortsarz and Peleg [KP94] were the first to consider relative bounds for spanners. They gave a greedy  $O(\log |E|/|V|)$ -approximation algorithm for the problem of finding a 2-spanner with the minimum number of edges. This was then extended to variants of 2-spanners, e.g. *client-server 2-spanner* [EP01] and *fault-tolerant 2-spanner* [DK11a, DK11b] (for which only  $O(\log \Delta)$  is known). All of these bounds are basically optimal, assuming  $P \neq NP$ , due to a hardness result of Kortsarz [Kor01].

## 1.5 Outline

We begin in Section 2 by giving a high-level overview of our reduction from LD2S to  $SmES$  and our faithful rounding for  $SmES$ . This overview is not technically accurate, but provides a simplified algorithm and analysis in order to provide the intuition behind our approach. In Section 3 we give a formal description of the LP relaxations for both problems. In Section 4 we give the details behind our reduction. Section 5 contains preliminaries for our  $SmES$  rounding algorithm, while the algorithm itself can be found in Section 6, and the analysis is in Section 7. Finally, we conclude with a discussion of future directions Section 8.

# 2 Overview of our approach

## 2.1 Overview of LP relaxation for LD2S and reduction to $SmES$

In this section we give an LP relaxation for LD2S that uses a relaxation of  $SmES$  as a black box, as well as an algorithm that shows how to use a faithful rounding for  $SmES$  to approximate LD2S. Both the relaxation and the algorithm presented here are simplifications that ignore some technical details; the full relaxation and algorithm, as well as all proofs, can be found in Sections 3 and 4.

We will actually give a relaxation for a slightly more general version of LD2S in which instead of spanning *all* edges we are given a subset  $\hat{E} \subseteq E$  and are only required to span edges in  $\hat{E}$ . Note that the optimal solution for demands  $\hat{E} \subseteq E$  has maximum degree that is at most the maximum

degree of the optimal solution to the original LD2S problem (where all edges are demands). This will allow us to cover some demands, re-solve the LP with only the remaining demands, and repeat.

Our relaxation is a feasibility LP, so we will guess the optimal degree bound  $\lambda$  and use it as a constant in the LP. For each  $u \in V$ , let  $G_u = (V_u, E_u)$  be the induced subgraph of  $(V, \hat{E})$  on  $\Gamma_G(u)$ . Our relaxation includes a fractional *smES* solution for each  $G_u$ : let **SmES-LP**( $G_u$ ) be a linear relaxation of *smES* with variables  $\{z_v^u\}_{v \in V_u} \cup \{z_e^u\}_{e \in E_u}$  with the property that  $z_{\{u,v\}}^u \leq \min\{z_w^u, z_v^u\}$  for all  $\{w, v\} \in E_u$ . In a 0-1 solution this means an edge is covered only if both of its endpoints are chosen. Any polytope that includes this basic condition can be used, but obviously the tighter this relaxation is the tighter our LD2S relaxation will be, and in the end we will use a much stronger relaxation for *smES* that is based on the Sherali-Adams hierarchy.

Our relaxation for LD2S with demands  $\hat{E} \subseteq E$  is given by (4)-(7).

$$((z_v^u)_{v \in V(G_u)}, (z_e^u)_{e \in E(G_u)}) \in \mathbf{SmES-LP}(G_u) \quad \forall u \in V \quad (4)$$

$$\max\{z_v^u, z_u^v\} \leq x_{\{u,v\}} \quad \forall \{u, v\} \in E \quad (5)$$

$$\sum_{v \in \Gamma(u)} x_{\{u,v\}} \leq \lambda \quad \forall u \in V \quad (6)$$

$$x_{\{u,v\}} + \sum_{w \in \Gamma(u) \cap \Gamma(v)} z_{\{u,v\}}^w = 1 \quad \forall \{u, v\} \in \hat{E} \quad (7)$$

Constraint (4) requires that for each neighborhood graph  $G_u$  there is an associated fractional *smES* solution.<sup>4</sup> Constraint (5) simply requires that for each edge, if either of the *smES* instances at its endpoints include it in their solution then we include it in the overall solution. Constraint (6) gives the degree bound, and (7) is the main covering constraint, requiring that every demand is either included or is spanned by a 2-path. It is easy to see that this is a valid relaxation for LD2S: if we are given a 2-spanner  $H$  of  $G$  with maximum degree at most  $\lambda$ , for every edge  $\{u, v\} \in E(H)$  we set  $x_{\{u,v\}} = 1$  and  $z_u^v = 1$  and  $z_v^u = 1$ . For every edge  $\{u, v\} \in E \setminus E(H)$  we arbitrarily choose some  $w \in V$  so that  $\{u, w\} \in E(H)$  and  $\{w, v\} \in E(H)$  (some such  $w$  must exist since  $H$  is a 2-spanner) and set  $z_{\{u,v\}}^w = 1$ . All other variables are 0.

We now show that it is sufficient to design a rounding scheme for *smES* that is faithful according to the following definition. Given a graph  $G$ , let  $\mathcal{L}(G)$  be an LP that has a variable  $\zeta_u$  for every  $u \in V(G)$  and a variable  $\zeta_e$  for every  $e \in E(G)$  (we will later instantiate  $\mathcal{L}(G)$  as various LP relaxations of *smES*).

**Definition 2.1.** A randomized rounding algorithm  $\mathcal{A}$  is a *factor  $f$  faithful rounding* for  $\mathcal{L}(G)$  if, when given a valid solution  $((\zeta_u)_{u \in V}, (\zeta_e)_{e \in E})$  to  $\mathcal{L}(G)$ , it produces a randomized (not necessarily induced) subgraph  $H^* = (V^*, E^*)$  such that

1.  $\Pr[v \in V^*] \leq f \cdot \zeta_v$  for all  $v \in V(G)$ ,
2.  $\Pr[\{u, v\} \in E^*] \leq \zeta_{\{u,v\}}$  for all  $\{u, v\} \in E(G)$ ,
3.  $|V^*| \leq f \cdot \sum_{v \in V(G)} \zeta_v$  (with probability 1), and
4.  $\mathbb{E}[|E^*|] \geq \tilde{\Omega}(\sum_{\{u,v\} \in E(G)} \zeta_{\{u,v\}})$ .

---

<sup>4</sup>Our actual relaxation (Figure 3) has a collection of *smES* instances for each neighborhood graph based on the possible degrees in a bipartite decomposition of an optimal solution, and we allow the LP to fractionally “guess” which of these instances to use.

Observe that if algorithm  $\mathcal{A}$  is a factor  $f$  faithful rounding for a relaxation of  $smES$  then it is also an  $f$ -approximation in the usual sense, simply by conditions 3 and 4 (up to a polylogarithmic loss in the amount of edges covered). The converse, however is not true: many rounding algorithms that give an  $f$ -approximation are not faithful, including [BCC<sup>+</sup>10].

We now show that if we are given an algorithm  $\mathcal{A}$  that is a factor  $f(n)$  faithful rounding for  $smES$  (where  $n$  is the number of vertices in the  $smES$  instance), there is an  $\tilde{O}(f(\Delta))$ -approximation algorithm for LD2S that uses algorithm  $\mathcal{A}$  as a black box. The reduction is given as Algorithm 1. It begins with all edges as the demand set  $\hat{E}$ , and first solves the LP relaxation for LD2S with demand set  $\hat{E}$ . It adds every edge that has  $x$  value at least  $1/4$ , and then uses algorithm  $\mathcal{A}$  to round each of the  $|V|$   $smES$  instances in the relaxation. At the end of the loop it updates the demands  $\hat{E}$  by removing edges that were successfully covered by this process, and repeats. Note that the *edges* covered by the  $smES$  roundings are used only in the analysis; in the algorithm we take the *vertices* output by each  $smES$  solution and include the appropriate edges in our spanner.

---

**Algorithm 1:** Approximation algorithm for LD2S

---

**Input** : Graph  $G = (V, E)$ , degree bound  $\lambda$ , factor  $f(n)$  faithful rounding algorithm  $\mathcal{A}$  for  $smES$

**Output:** 2-spanner  $H = (V, E_H)$  of  $G$

- 1  $\hat{E} \leftarrow E, E_H \leftarrow \emptyset$
- 2 **while**  $\hat{E} \neq \emptyset$  **do**
- 3     Compute a valid solution  $\langle \vec{x}, \vec{z} \rangle$  for LP (4)-(7) on graph  $G$  with demands  $\hat{E}$
- 4      $E_x \leftarrow \{e \in E : x_e \geq 1/4\}$
- 5     **foreach**  $u \in V$  **do**
- 6          $H_u^* \leftarrow \mathcal{A}(G_u, \vec{z}^u)$ ;   // output of  $smES$  rounding  $\mathcal{A}$
- 7          $E_u \leftarrow \{\{u, v\} \in E : v \in V(H_u^*)\}$
- // Add all edges found in above rounding
- 8      $E_H \leftarrow E_H \cup E_x \cup (\bigcup_{u \in V} E_u)$
- // Remove satisfied demands
- 9      $\hat{E} \leftarrow \hat{E} \setminus (E_H \cup \{\{u, v\} : \exists w \in V \text{ s.t. both } \{u, w\}, \{w, v\} \in E_H\})$

---

**Theorem 2.2.** *Let algorithm  $\mathcal{A}$  be a factor  $f(n)$  faithful rounding for  $smES$  (where  $n$  is the number of vertices in the  $smES$  instance). Then there is a (randomized)  $\tilde{O}(f(\Delta))$ -approximation for LD2S.*

*Proof.* We provide only a sketch of the proof; details can be found in Section 4. We may assume that our LD2S algorithm guesses some  $\lambda \in [\text{OPT}, 2 \cdot \text{OPT}]$  simply by trying the  $O(\log \Delta)$  relevant values and reporting the best solution. In this case, LP (4)-(7) is guaranteed to have a feasible solution. We now use Algorithm 1 with this value of  $\lambda$ . It is easy to see that each iteration of the loop only increases the maximum degree by  $\tilde{O}(f(\Delta)) \cdot \text{OPT}$ : adding edges in  $E_x$  only costs a constant factor more than the fractional solution, Definition 2.1(3) implies that rounding the  $smES$  solution at  $u$  only increases the degree of  $u$  by  $f(|V_u|) \cdot \text{OPT} \leq f(\Delta) \cdot \text{OPT}$ , and Definition 2.1(1) implies that rounding the  $smES$  solution at neighbors of  $u$  only increases the degree of  $u$  by  $\tilde{O}(f(|V_u|)) \cdot \text{OPT}$  (with high probability).

So we just need to show that the number of iterations is (with high probability) at most  $\tilde{O}(1)$ . To do this we prove that in every iteration the expected number of satisfied demands is at least



$\tilde{\Omega}(|\hat{E}|)$ . This is clearly true if  $|E_x|$  is large. If  $|E_x|$  is small, then summing (7) over all  $\{u, v\} \in \hat{E}$  implies that the total amount of demand covered by *smES* instances (i.e. the  $z$  variables) is large, say  $\Omega(|\hat{E}|)$ . Now Definition 2.1(4) guarantees that when we round an instance we still cover almost as much demand as the LP (up to a polylogarithmic factor), and Definition 2.1(2) implies that this coverage is spread out among the demands in a way that corresponds to the LP. This, together with (7), implies that the total coverage is large but no edge is “overcovered”. Thus while we cannot guarantee that *every* demand is covered with *high* probability in each iteration, an averaging argument guarantees that *many* demands are covered with *reasonable* probability. This is enough to give the expected amount of coverage that we need.  $\square$

## 2.2 Overview of our faithful rounding algorithm for *smES*

In this section we describe our faithful factor  $n^{3-2\sqrt{2}+\varepsilon}$  rounding algorithm for *smES*. While the description of the full algorithm is rather lengthy, and therefore deferred to Sections 5, 6, and 7, we give a high-level overview (with some technical details) and concentrate on a special case which illustrates the main ideas in the algorithm and its analysis.

### 2.2.1 DENSEST $k$ -SUBGRAPH and the log-density framework

We follow the framework introduced in [BCC<sup>+</sup>10]. They begin by defining the notion of log-density of a graph as  $\log_n(D_{\text{avg}})$ , where  $D_{\text{avg}}$  is the average degree and  $n$  is the number of nodes. They then asked the following question: how hard is it to distinguish between 1) a random graph, and 2) a graph containing a subgraph with roughly the same log-density as the first graph?

More formally, they pose the following DENSE VERSUS RANDOM promise problem, parameterized by  $k$  and constants  $0 < \alpha, \beta < 1$ : given a graph  $G$ , distinguish between the following two cases:

1.  $G = G(n, p)$  where  $p = n^{\alpha-1}$  (this graph has log-density concentrated around  $\alpha$ ).
2.  $G$  is adversarially chosen so that the densest  $k$ -subgraph has log-density  $\beta$  (where  $k^{1+\beta} \gg pk$ ).

For certain ranges of parameters, it seems quite challenging to efficiently distinguish when  $\beta < \alpha$ . In fact, the following hypothesis is consistent with the current state of our knowledge:

**Hypothesis 2.3.** *For all  $0 < \alpha < 1$ , for all sufficiently small  $\varepsilon > 0$ , and for all  $k \leq \sqrt{n}$ , we cannot solve DENSE VERSUS RANDOM in polynomial time (w.h.p.) when  $\beta \leq \alpha - \varepsilon$ .*

The above hypothesis (if true) has immediate implications for the hardness of approximation of both *Dks* and *smES*. Concretely, for *smES*, let  $m = k^{1+\beta}$  be the number of edges in  $k$ -subgraph in the second case. We know that in the first case w.h.p. the smallest  $m$ -edge subgraph has size at least  $\tilde{\Omega}(\min\{m, \sqrt{mn^{1-\alpha}}\})$ . Thus, if we could achieve approximation ratio  $\ll k / \min\{m, \sqrt{mn^{1-\alpha}}\}$ , this would refute Hypothesis 2.3 for the corresponding parameters. For  $k = n^{\sqrt{2}-1}$  and  $\alpha = \sqrt{2} - 1$ , the hypothesis implies that there exists no  $n^{3-2\sqrt{2}-\varepsilon}$ -approximation for *smES*.

While [BCC<sup>+</sup>10] matches the gap predicted by the log-density model for *Dks* with an  $n^{1/4+\varepsilon}$  approximation for *Dks* (even for general graphs), we also match the predicted gap for *smES* with an  $n^{3-2\sqrt{2}+\varepsilon}$ -approximation for *smES*.

### 2.2.2 Parametrization and simplifications

In order to achieve a faithful rounding, we will make certain assumptions (which we later justify) about the structure of the intended solution to the LP relaxation. In particular, we will assume that the subgraph represented by the solution is regular, and that we are allowed to “guess” the size of the subgraph,  $k$ , and the degree in the subgraph,  $d$ , thus  $m = \Theta(kd)$  (see Section 3.1).

We also make the following simplifying assumptions. Let  $f = f(n, k, d)$  be the intended approximation factor, which will be determined shortly. We may assume that  $f \leq d$ , since it is easy to achieve a faithful  $O(d)$ -approximation (see Appendix A). We also assume that the maximum degree in the input graph is at most  $D = nd/(kf^2)$  (see Appendix B).

Finally write  $\alpha = \log_n(D)$ , and define our intended approximation  $f$  implicitly as the value which satisfies  $f = n^{\alpha(1-\alpha)/(1+\alpha)}$  (together with the definition of  $D$  we can derive an explicit expression for  $\alpha$  and  $f$ ). Note that maximizing this expression over  $\alpha \in [0, 1]$  shows that  $f \leq n^{3-2\sqrt{2}}$ .

### 2.2.3 LP relaxation and faithful rounding for $SmES$

With the previous assumptions in mind, we have the following feasibility-LP relaxation (simplified for this overview) which is implied by  $q$  rounds of Sherali-Adams, with variables  $\{z_T \mid T \subset V \cup E, |T| \leq q\}$  (in the intended 0-1 solution,  $z_T = 1$  if and only if all vertices and edges in  $T$  are in the subgraph):

$$\sum_{v \in V} z_{T \cup \{v\}} = kz_T \quad \forall T \subset V \cup E, |T| \leq q - 1 \quad (8)$$

$$\sum_{u \in \Gamma(v)} z_{T \cup \{u\}} = dz_T \quad \forall T \subset V \cup E, |T| \leq q - 1, \forall v \in T \cap V \quad (9)$$

$$z_T = z_{T \cup \{u\}} = z_{T \cup \{v\}} = z_{T \cup \{u, v\}} \quad \forall T \subset V \cup E, |T| \leq q - 2, \forall \{u, v\} \in T \cap E \quad (10)$$

$$0 \leq z_T \leq z_{T'} \leq z_\emptyset = 1 \quad \forall T' \subseteq T \quad (11)$$

The algorithm in its full generality is based on the caterpillar structures introduced in [BCC<sup>+</sup>10] (where the caterpillar structure depends on  $\alpha$ ). Let us concentrate here on the case where  $\alpha = 1/s$  for some (fixed) integer  $s > 0$ , in which case the caterpillar is simply a path of length  $s$ . At its core, the algorithm (for this value of  $\alpha$ ) relies on an LP-analogue of the following combinatorial argument. Fix a vertex  $v_0$  in the optimum subgraph. For all  $t = 1, \dots, s$ , let  $P_t^{v_0}$  be the union of all (possibly self-intersecting) paths of length  $t$  in the subgraph starting at  $v_0$ , and let  $V_t^{v_0}$  be final endpoints of those paths. Note that  $|V_1^{v_0}| = d$  and that  $|V_s^{v_0}| \leq k = \frac{dn}{f^2 D} = \frac{d}{f^2} \cdot n^{1-\alpha} = \frac{d}{f^2} f^{(1+\alpha)/\alpha} = df^{s-1}$ . Therefore, there must be some  $t \in \{1, \dots, s-1\}$  for which  $|V_{t+1}^{v_0}|/|V_t^{v_0}| \leq f$ . Now consider the subgraph at this step  $H_t^{v_0} = (V_t^{v_0}, V_{t+1}^{v_0}, \{\{v_t, v_{t+1}\} \mid \exists v_0 - \dots - v_t - v_{t+1} \in P_{t+1}^{v_0}\})$ . Since the vertices in  $V_t^{v_0}$  all have degree  $d$ , the average degree of vertices in  $V_{t+1}^{v_0}$  is at least  $d/f$ . It turns out that even without access to the optimum subgraph we can isolate a subgraph with average degree at least  $d/f$  and at most  $kf$  vertices (this is essentially because by the degree bound, the number of vertices at any intermediate stage is at most  $D^{s-1} = n^{1-\alpha} = \frac{n}{D} = k \cdot \frac{f^2}{d} \leq kf$ ). This essentially gives an  $f$ -approximation for  $SmES$  (since we can repeat until accumulating  $m$  edges).

Here we come to the fundamental difficulty in adapting such an approach to achieve a faithful rounding. The combinatorial algorithm depends on choosing an initial vertex  $v_0$  which is actually in the optimum subgraph. The analogous LP-rounding algorithm uses the LP values “conditioned

on choosing  $v_0$ ”, that is, values of the form  $z_{S \cup \{v_0\}}/z_{v_0}$  instead of the original  $z_S$  variables (where  $S$  corresponds to one or more vertices/edges along the path). However, it is the  $z_S$  variables (in particular for singleton sets  $S$  representing one vertex or one edge) which we want to be faithful to in our rounding.<sup>5</sup> Unfortunately, these two LP solutions might be almost completely unrelated.

To overcome this difficulty, we use a somewhat elaborate bucketing scheme, to ensure that all the relevant LP values are reasonably uniform, as follows. Denote by  $\mathcal{P}_t^v$  the set of all length  $t$  paths in the graph starting at vertex  $v$ , and by  $z_p$  the variable for a path  $p$  (i.e.,  $z_T$  where  $T$  is the set of edges and vertices in  $p$ , or by Constraint (11),  $T$  could equivalently be just the edges in  $p$ ). The core of the analysis of the LP-analogue relies on the equality

$$\sum_v \sum_{p \in \mathcal{P}_t^v} z_p = \sum_v d^t z_{\{v\}} = kd^t,$$

obtained by Constraint (8) and repeated applications of (9), but in fact it can use any set of length- $s$  paths  $\mathcal{P}$  for which  $\sum_{p \in \mathcal{P}} z_p = \tilde{\Omega}(kd^s)$ . Thus by partitioning the set of paths  $\bigcup_v \mathcal{P}_s^v$  into buckets and choosing a bucket  $\mathcal{P}$  with the largest LP value, we can ensure that in every path  $p = u_0 - u_1 - \dots - u_s$  in the bucket  $\mathcal{P}$  certain LP values (like the ones corresponding to entire paths,  $z_p$ , or the ones corresponding to path prefixes,  $z_{\{u_{i-1}, u_i\} | i \in [t]}$  for some  $t \in [s-1]$ , or to vertices in certain positions,  $z_{\{u_t\}}$ , or to “conditioned” values,  $z_{\{u_0, u_t\}}/z_{\{u_0\}}$ ) are all independent of the choice of path (up to a constant factor). In other words, within the bucket  $\mathcal{P}$  (say, vertices  $u_t$  for a fixed  $t \in \{0, \dots, s\}$ ), the corresponding LP values will be essentially uniform over the choice of starting vertex  $u_0$  and path  $p$ .

Using the uniformity obtained via the above bucketing scheme, we can relate the algorithm (which is based on the conditioned LP values) to the original LP values. After some additional combinatorial bucketing, we can run the following algorithm: let  $\mathcal{V}_0$  be the set of starting vertices  $u_0$  (i.e. paths of length 0) that survive the bucketing, pick a starting vertex  $u_0 \in \mathcal{V}_0$  uniformly at random, and for whichever level  $t \in [s-1]$  that gives the approximation guarantee (it can be shown that such a  $t$  exists), output the level  $t$  subgraph  $H_t^{u_0} = \{\{u_t, u_{t+1}\} \mid \exists p = u_0 - u_1 - \dots - u_s \in \mathcal{P}\}$ . Since LP values are uniform, the question essentially becomes, how do we guarantee that no bucketed vertex (or edge) is chosen with much higher probability than the rest (or the average)? This is where we crucially use the regularity Constraint (9) (as opposed to, say, a minimum degree constraint, as in [BCC<sup>+</sup>10]). Roughly speaking, individual vertices and edges cannot be reached by a disproportionately large fraction of vertices  $u_0 \in \mathcal{V}_0$ , because then the relative total LP weight of the corresponding paths (to such a vertex or edge) would exceed  $d^t$ .

For the sake of concreteness, let us consider one specific aspect of faithful rounding: the probability with which the level  $t$  vertices  $u_t$  are chosen. Let  $\mathcal{P}_t$  be the set of length  $t$  prefixes of paths in  $\mathcal{P}$ , let  $\mathcal{P}_t^{u_0}$  be the set of paths in  $\mathcal{P}_t$  that start with the vertex  $u_0 \in \mathcal{V}_0$ , and let  $\mathcal{V}_t$  (resp.  $\mathcal{V}_t^{u_0}$ ) be the set of level  $t$  endpoints of paths in  $\mathcal{P}_t$  (resp. in  $\mathcal{P}_t^{u_0}$ ). By the approximation guarantee (via an LP analogue of the above combinatorial argument), we have

$$|\mathcal{V}_t^{u_0}| \leq fk. \tag{12}$$

Suppose the bucketing also ensures that every  $u_0 \in \mathcal{V}_0$  and  $u_t \in \mathcal{V}_t^{u_0}$  are connected by roughly the same number of  $\mathcal{P}_t$  paths (up to a constant factor), which we denote by  $h$ . Also, suppose the cardinalities  $|\mathcal{V}_t^{u_0}|$  are roughly uniform for different choices of  $u_0$ . Then, abusing notation, we can

---

<sup>5</sup>This problem is only exacerbated in the general case, when the caterpillar has additional leaves to condition on.

write the number of paths as  $|\mathcal{P}_t| \approx |\mathcal{V}_0| \cdot |\mathcal{V}_t^{u_0}| \cdot h$ , and in particular, the total weight of paths  $p \in \mathcal{P}_t$  is  $z_p |\mathcal{V}_0| \cdot |\mathcal{V}_t^{u_0}| \cdot h \approx kd^t$ . Now, by repeated applications of Constraint (9), we have that the total weight of paths leading to a specific vertex  $u_t \in \mathcal{V}_t$  is  $z_p |\{u_0 \mid u_t \in \mathcal{V}_t^{u_0}\}| h \leq z_{u_t} d^t$  (note that this argument reverses the direction of paths in the algorithm and so crucially depends on the existence of consistent high-moment Sherali-Adams variables, which are not present in the Lovász-Schrijver hierarchy used in [BCC<sup>+</sup>10]). Combining this with the (approximate) equality above, we can bound the probability that a vertex  $u_t$  is included in the output (the level  $t$  subgraph) as

$$\frac{|\{u_0 \mid u_t \in \mathcal{V}_t^{u_0}\}|}{|\mathcal{V}_0|} \leq \frac{d^t z_{u_t}}{z_p h |\mathcal{V}_0|} = \frac{|\mathcal{V}_t^{u_0}| z_{u_t}}{k} \leq f z_{u_t}, \text{ by (12).}$$

### 3 LP relaxations

In this section we develop the basic LP relaxations that we will use, both for *smES* and for LD2S. We begin with *smES*, since we will need the LP we develop in order to define the LP for LD2S.

#### 3.1 LP relaxation for *smES*

It turns out that it is easier to develop faithful rounding algorithms for *smES* if we make certain simplifying assumptions. Namely, we would like to assume that the input graph is bipartite, and that the optimal solution is nearly-regular (vertices on the same side of the bipartition have degree within an  $O(\log n)$  factor of each other). These assumptions will affect our relaxation, so we discuss them here. Since these assumptions involve manipulations of the optimal (or at least an unknown) subgraph, one should view these as a thought experiment which justifies the correctness (i.e. feasibility) of our relaxations.

Formally, we define nearly-regular as follows:

**Definition 3.1.** A bipartite graph  $G = (U_0, U_1, E)$  is called  $(k_0, k_1, d_0, d_1)$ -*nearly regular* if for every  $b \in [2]$  we have  $|U_b| = k_b$  and the following condition on the degrees holds:

$$d_b \geq \max_{u_b \in U_b} \deg(u_b) \geq \min_{u_b \in U_b} \deg(u_b) \geq \Omega(d_b / \log n)$$

By convention, we will assume that  $k_0 \geq k_1$ , which implies that  $d_1 \geq \Omega(d_0 / \log n)$ .

The next lemma shows that any graph  $H$  can be changed into a nearly-regular graph with almost the same number of edges. In particular, any dense subgraph  $H$  can be made nearly regular without losing too much in the density.

**Lemma 3.2.** *Given an arbitrary graph  $H = (V, E)$  on  $n$  vertices, there exist values  $k_0, k_1, d_0, d_1$  and disjoint vertex sets  $U_0, U_1 \subseteq V$  with the properties that the induced bipartite subgraph  $H'$  of  $H$  on  $(U_0, U_1)$  is  $(k_0, k_1, d_0, d_1)$ -nearly regular and also has  $|E(U_0, U_1)| \geq \Omega(1/\log^2 n)|E|$ .*

*Proof.* First, note that a random cut in the graph gives a bipartition which preserves at least half the edges (in expectation). Thus there exists a bipartition  $(V_0, V_1)$  with  $|E(V_0, V_1)| \geq |E|/2$ . For the rest of the proof we will only use these edges between  $V_0$  and  $V_1$ . Now, partition the vertices in  $V_0$  into  $\log n$  buckets by degree (into  $V_1$ ), where in each bucket, degrees vary up to a factor of at most 2. Let  $\tilde{U}_0$  be the bucket which sees the most edges (at least  $|E|/2 \log n$ ). Let  $d_0$  be the maximum degree in  $\tilde{U}_0$ . Note that the average degree in  $\tilde{U}_0$  is at least  $d_0/2$ .

Now partition the vertices of  $V_1$  into buckets by their degree into  $\tilde{U}_0$ . Let  $\tilde{U}_1$  be the bucket with the largest number of edges into  $\tilde{U}_0$  (at least  $|E|/2 \log^2 n$ ). Let  $d_1$  be the maximum degree (into  $\tilde{U}_0$ ) in  $\tilde{U}_1$ , and note that the average degree in  $\tilde{U}_1$  is at least  $d_1/2$ . Since the number of edges in this step went down by a most a  $\log n$  factor, the average degree in  $\tilde{U}_0$  (into  $\tilde{U}_1$ ) is at least  $d_0/(2 \log n)$ . If we now iteratively remove every vertex in  $\tilde{U}_0$  with degree (into  $\tilde{U}_0$ ) at most  $d_0/(6 \log n)$ , and every vertex in  $\tilde{U}_1$  with degree (into  $\tilde{U}_0$ ) at most  $d_1/6$ , then it is easy to see that at least a  $1/3$ -fraction of the original edges in  $E(\tilde{U}_0, \tilde{U}_1)$  remains, and this final step guarantees the regularity conditions on both sides.  $\square$

Before defining our relaxation for general graphs, let us first define a feasibility LP relaxation for the decision problem of whether, given a bipartite graph  $G' = (V_0, V_1, E)$ , the graph contains a  $(k_0, k_1, d_0, d_1)$ -nearly regular subgraph, with  $k_0$  vertices on the  $V_0$  side, and  $k_1$  vertices on the  $V_1$  side. For any integer  $q$ , let  $\mathcal{T}_q = \mathcal{T}_q(G') = \{T \subseteq V_0 \cup V_1 \cup E' : |T| \leq q\}$ .

We denote by **Bipartite-SmES-LP** $_q(G', k_0, k_1, d_0, d_1)$  the set of solutions to a feasibility LP given by Constraints (13)-(16), over the variables  $(y_T)_{T \in \mathcal{T}_q}$ , as depicted in Figure 1. These constraints are actually implied by  $q$  rounds of Sherali-Adams applied to a basic *smES* LP that has variables for all vertices *and* edges.

---

Figure 1: Relaxation **Bipartite-SmES-LP** $_q(G', k_0, k_1, d_0, d_1)$  on variables  $(y_T)_{T \in \mathcal{T}_q}$

---

$$\sum_{v \in V_b} y_{T \cup \{v\}} = k_b y_T \quad \forall b \in \{0, 1\}, \forall T \in \mathcal{T}_{q-1} \quad (13)$$

$$\Omega\left(\frac{1}{\log n}\right) d_b y_T \leq \sum_{u \in \Gamma(v)} y_{T \cup \{u, v\}} \leq d_b y_T \quad \forall b \in \{0, 1\}, \forall T \in \mathcal{T}_{q-1}, \forall v \in T \cap V_b \quad (14)$$

$$y_T = y_{T \cup \{u\}} = y_{T \cup \{v\}} = y_{T \cup \{u, v\}} \quad \forall T \in \mathcal{T}_{q-2}, \forall \{u, v\} \in T \quad (15)$$

$$0 \leq y_T \leq y_{T'} \leq 1 \quad \forall T' \subseteq T \in \mathcal{T}_q \quad (16)$$


---

**Remark.** Normally, such an LP also includes the normalization  $y_\emptyset = 1$ . However, for our intended usage of this LP, namely for LD2S, it will be important to leave this variable unconstrained, except for the upper bound given by (16). But when we round this LP we will be able to assume that  $y_\emptyset = 1$ ; whenever we choose to round it we will also scale it by  $1/y_\emptyset$ . Thus when we consider faithful rounding algorithms for this LP we will always be assuming that  $y_\emptyset = 1$ .

**Remark.** This LP has size (variables and constraints) at most  $n^{O(q)}$ , because the number of variables is dominated by  $O(|\mathcal{T}_q|)$ , and the number of constraints is dominated by  $O(|\mathcal{T}_q|^2)$ .

Notice that according to Constraint (15), an edge is covered if and only if both endpoints are chosen *and* we decide to take the edge. For example, it is feasible to have  $y_{\{u\}} = y_{\{v\}} = y_{\{u, v\}} = 1$  (i.e. both  $u$  and  $v$  are chosen) but  $y_{\{\{u, v\}\}} = y_{\{u, v, \{u, v\}\}} = 0$  (i.e. for some reason the LP does not count this edge as being covered). Normally for covering problems such as *smES* there is no reason not to include an edge if both vertices are included, but because of how we use *smES* in our algorithm for LD2S it will be important for us to be able to include two vertices without necessarily covering the edge between them.

Observe that if  $G'$  contains a  $(k_0, k_1, d_0, d_1)$ -nearly regular subgraph  $H$ , then the above LP has a feasible solution that corresponds to  $H$  in the sense that  $y_T = 1$  if and only if every vertex and edge in  $T$  is present in  $H$  (and  $y_\emptyset = 1$ ).

Now, consider a (not necessarily bipartite) graph  $G$ . Lemma 3.2 implies that for any subgraph  $H$  of  $G$  there is a nearly-regular bipartite subgraph of  $H$  with almost as many edges. However, since we do not know a priori the bipartition of the subgraph (or any bipartition of  $G$  which is consistent with it), we write a “container” LP whose purpose is to essentially assign sides, and to interface cleanly between the bipartite *smES* relaxation, and the relaxation for LD2S. In brief, with every vertex  $v \in V$ , we associate at most one of two assignments, which we label  $(v, 1)$  and  $(v, 2)$ . Thus, every edge  $\{u, v\}$  can participate in the subgraph as (at most) one of two possible “assigned” edges  $\{(u, 1), (v, 2)\}$  and  $\{(u, 2), (v, 1)\}$ . We can think of these as forming a bipartite graph  $B(G)$  with vertices  $V \times [2]$  and edges  $\{(u, 1), (v, 2)\}$  and  $\{(u, 2), (v, 1)\}$  for every  $\{u, v\} \in E$ . The induced nearly-regular bipartite subgraph  $H'$  of  $G$  that is guaranteed to exist by Lemma 3.2 corresponds to an induced subgraph of  $B(G)$  in which for every vertex  $v$  in  $H'$  either  $(v, 1)$  or  $(v, 2)$  is included (depending on the side of  $v$  in  $H'$ ). Note that for every vertex (or edge) in  $H'$ , there is exactly one vertex (or edge) in the corresponding subgraph of  $B(G)$ .

It will be easier to describe our algorithm and analysis for *smES* using the LP for bipartite graphs, but in order to interface with the LD2S LP we will use the above discussion of  $B(G)$  to write a container or wrapper LP. Denote by  $\mathbf{SmES-LP}_q(G, k_0, k_1, d_0, d_1)$  the set of solutions to the feasibility LP depicted in Figure 2 on the variables  $(z_a)_{a \in V \cup E}$  and  $z_\emptyset$  (in other words there is a  $z$  variable for every vertex and edge in the original graph, as well as one extra  $z$  variable for the empty set). Notice that this LP has auxiliary variables of the form  $y_T$ .

---

Figure 2: Relaxation  $\mathbf{SmES-LP}_q(G, k_0, k_1, d_0, d_1)$  on variables  $(z_a)_{a \in V \cup E \cup \{\emptyset\}}$

---

$$\exists (y_T)_{T \in \mathcal{T}_q(B(G))} \in \mathbf{Bipartite-SmES-LP}_q(B(G), k_0, k_1, d_0, d_1) \text{ s.t.}$$

$$z_\emptyset = y_\emptyset \tag{17}$$

$$y_{\{(u,1)\}} + y_{\{(u,2)\}} = z_u \leq z_\emptyset \quad \forall u \in V \tag{18}$$

$$y_{\{(u,1),(v,2)\}} + y_{\{(u,2),(v,1)\}} = z_{\{u,v\}} \leq z_u, z_v \quad \forall \{u, v\} \in E \tag{19}$$


---

It is easy to see that for any  $(k_0, k_1, d_0, d_1)$ -nearly regular bipartite subgraph  $H$  of  $G$ , if we set  $z_u = 1$  for  $u \in V(H)$  and  $z_e = 1$  for  $e \in E(H)$  then we can set the  $y$  variables in a way corresponding to the associated subgraph of  $B(G)$ , giving a valid solution. For any tuple of parameters  $\tau = \langle k_0, k_1, d_0, d_1 \rangle$  let  $\mathbf{SmES-LP}_q^\tau(G) = \mathbf{SmES-LP}_q(G, k_0, k_1, d_0, d_1)$  and let  $\mathbf{Bipartite-SmES-LP}_q^\tau(G') = \mathbf{Bipartite-SmES-LP}_q(G', k_0, k_1, d_0, d_1)$ . By construction, a faithful rounding for  $\mathbf{Bipartite-SmES-LP}_q^\tau(B(G))$  (according to Definition 2.1) implies a faithful algorithm for  $\mathbf{SmES-LP}_q^\tau(G)$ ; we now prove this.

**Lemma 3.3.** *For any graph  $G$  and parameters  $\tau$ , if we have a factor  $f$  faithful rounding algorithm for  $\mathbf{Bipartite-SmES-LP}_q^\tau(B(G))$  then we have a factor  $f$  faithful rounding algorithm for  $\mathbf{SmES-LP}_q^\tau(G)$ .*

*Proof.* Let algorithm  $\mathcal{A}$  be a factor  $f$  faithful rounding for  $\mathbf{Bipartite-SmES-LP}_q^\tau(B(G))$ . Then our algorithm for rounding  $\mathbf{SmES-LP}_q^\tau(G)$  is simple: we first find the associated  $y$  variables (if

they are not already given to us), and then run algorithm  $\mathcal{A}$  on the associated  $y$  variables to get  $(\tilde{V}^*, \tilde{E}^*)$ . We then include a vertex  $v$  in  $V^*$  if  $\tilde{V}^*$  includes either  $(v, 1)$  or  $(v, 2)$ , and include an edge  $\{u, v\}$  in  $E^*$  if  $\tilde{E}^*$  includes either  $\{(u, 1)(v, 2)\}$  or  $\{(u, 2), (v, 1)\}$ .

Then  $\Pr[v \in V^*] \leq f \cdot y_{\{(v,1)\}} + f \cdot y_{\{(v,2)\}} = f \cdot z_v$ , satisfying the first part of the definition. And  $\Pr[\{u, v\} \in E^*] \leq y_{\{(u,1),(v,2)\}} + y_{\{(u,2),(v,1)\}} = z_{\{u,v\}}$ , satisfying the second part. For the third part, we know that  $|\tilde{V}^*|$  is (with probability 1) at most  $f \cdot \sum_{v \in V(G)} (y_{\{(v,1)\}} + y_{\{(v,2)\}}) = f \cdot \sum_{v \in V(G)} z_v$  vertices, and clearly  $|V^*| \leq |\tilde{V}^*|$ . Finally, we know that  $\mathbb{E}[|E^*|] \geq \mathbb{E}[|\tilde{E}^*|]/2 \geq \tilde{\Omega}(\sum_{\{u,v\} \in E(G)} (y_{\{(u,1),(v,2)\}} + y_{\{(u,2),(v,1)\}})) = \tilde{\Omega}(\sum_{\{u,v\} \in E(G)} z_{\{u,v\}})$   $\square$

### 3.2 LP relaxation for LD2S

Now we show how to use this *smES* LP to give an LP relaxation for LD2S. We will actually give a relaxation for a slightly more general version of LD2S in which instead of spanning *all* edges we are given a subset  $\hat{E} \subseteq E$  and are only required to span edges in  $\hat{E}$ . Note that the optimal solution for demands  $\hat{E} \subseteq E$  has maximum degree that is at most the maximum degree of the optimal solution to the original LD2S problem (where all edges are demands). This will allow us to cover some demands, re-solve the LP with only the remaining demands, and repeat.

Since we construct a feasibility LP we will guess the optimal degree bound  $\lambda$ , so we will be able to treat it like a constant (rather than a variable). For each  $u \in V$ , let  $G_u = (V_u, E_u)$  be the induced subgraph of  $(V, \hat{E})$  on  $\Gamma_G(u)$ , i.e.  $G_u$  has vertex set  $\Gamma_G(u)$  (the neighbors of  $u$  in the overall graph) and edge set  $\hat{E}$  restricted to  $\Gamma_G(u)$ . The core of our relaxation is a decomposition of one *smES* instance into many *smES* instances. In particular, the following lemma will be useful:

**Lemma 3.4.** *Given a graph  $G = (V, E)$  on  $n$  vertices and a value  $\lambda$ , there is a multiset  $L^\lambda$  of size at most  $O(n^4 \log^3 n)$  consisting of tuples  $\langle k_0, k_1, d_0, d_1 \rangle$  so that every subgraph of  $G$  with at most  $\lambda$  vertices can be decomposed into  $O(\log^3 n)$  nearly-regular subgraphs with parameters from  $L^\lambda$ .*

*Proof.* Let  $H$  be a subgraph of  $G$  with at most  $\lambda$  vertices. We know from Lemma 3.2 (applied to  $H$ ) that there exists some  $\tau = \langle k_0, k_1, d_0, d_1 \rangle$  and subgraph  $H'$  of  $H$  so that  $H'$  is nearly-regular with parameters  $\tau$  and the number of edges in  $H'$  is at least an  $\Omega(1/\log^2 n)$ -fraction of the number of edges in  $H$ . We can now remove the edges in  $H'$  from  $H$  and repeat this process. Since there are at most  $n^2$  edges initially, we can repeat this step only  $O(\log^3 n)$  times. Note that this bound is independent of the graph  $H$  that we are decomposing.  $H$  simply affects *which* tuples are produced by the decomposition. But no matter what  $H$  is, obviously  $k_0, k_1, d_0, d_1$  are all at most  $\lambda \leq n$ . Thus there are at most  $n^4$  distinct tuples. So we simply define the multiset  $L^\lambda$  to contain  $O(\log^3 n)$  copies of each such tuple, for a total size of  $O(n^4 \log^3 n)$ .  $\square$

This decomposition lemma will allow us to cover *all* demands in our relaxation, even using the nearly-regular assumption in the *smES* relaxation. More formally, for LD2S we have the feasibility LP depicted in Figure 3 with the degree bound  $\lambda$  being treated as a constant.

**Remark.** This LP has size (variables and constraints)  $n^{O(q)}$ , so can be solved in polynomial time for constant  $q$ . Indeed, for each  $u \in V$  there are  $|L^\lambda| = \tilde{O}(n^4)$  different **SmES-LP** $_q$  programs. Each such program has  $n^{O(q)}$  variables and constraints, and in addition there are  $O(n^2)$  variables of the form  $x_{\{u,v\}}$  and  $O(n^2)$  new constraints (21)-(25).

---

Figure 3: Relaxation  $\mathbf{LD2S}_q^\lambda(G, \hat{E})$

---

$$(z_\emptyset^{u,\tau}, (z_v^{u,\tau})_{v \in V(G_u)}, (z_e^{u,\tau})_{e \in E(G_u)}) \in \mathbf{SmES-LP}_q^\tau(G_u) \quad \forall u \in V, \forall \tau \in L^\lambda \quad (20)$$

$$\sum_{\tau \in L^\lambda} z_\emptyset^{u,\tau} \leq O(\log^3 \Delta) \quad \forall u \in V \quad (21)$$

$$\sum_{\tau \in L^\lambda} z_v^{u,\tau} + \sum_{\tau \in L^\lambda} z_u^{v,\tau} \leq O(\log^3 \Delta) \cdot x_{\{u,v\}} \quad \forall \{u,v\} \in E \quad (22)$$

$$\sum_{v \in \Gamma(u)} x_{\{u,v\}} \leq \lambda \quad \forall u \in V \quad (23)$$

$$x_{\{u,v\}} + \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} z_{\{u,v\}}^{w,\tau} = 1 \quad \forall \{u,v\} \in \hat{E} \quad (24)$$

$$0 \leq x_{\{u,v\}} \leq 1 \quad \forall \{u,v\} \in E \quad (25)$$


---

**Lemma 3.5.** *The feasibility LP (20)-(25) is a valid relaxation of LD2S with degree bound  $\lambda$  and demands  $\hat{E} \subseteq E$ .*

*Proof.* Let  $G = (V, E)$  be a graph, and let  $H = (V, E_H)$  be a subgraph of  $G$  that is a valid 2-spanner and has maximum degree  $\Delta \leq \lambda$ . We construct an LP solution as follows. For each  $\{u, v\} \in E$ , set  $x_{\{u,v\}} = 1$  if  $\{u, v\} \in E_H$  and set  $x_{\{u,v\}} = 0$  otherwise. Note that Constraint (23) is satisfied. For every edge  $\{u, v\} \in E \setminus E_H$ , there is at least one 2-path between  $u$  and  $v$  in  $H$ , since  $H$  is a valid 2-spanner. Let  $w(u, v)$  be the center vertex of such a path, choosing one arbitrarily if there are multiple 2-paths. For every vertex  $w \in V$ , we define the graph  $H_w$  to be the subgraph of  $G_w$  with vertex set  $\{u \in \Gamma(w) : \{u, w\} \in E_H\}$  and all edges  $\{u, v\}$  with the property that  $w(u, v) = w$  (note that this also implies that  $x_{\{u,v\}} = 0$ ). We know from Lemma 3.4 that  $H_w$  can be decomposed into  $O(\log^3 \Delta)$  nearly-regular graphs with parameters from  $L^\lambda$  (here  $\Delta$  has replaced  $n$  because  $H_w$  has only  $\Delta$  vertices). Let the multiset  $L'_w \subseteq L^\lambda$  contain the parameter tuples  $\tau$  used in this decomposition, and for  $\tau \in L'_w$  let  $H_w^\tau$  be the graph from the decomposition. Then for each  $\tau \in L'_w$  we set  $z_\emptyset^{w,\tau} = 1$ , and we set  $z_u^{w,\tau} = 1$  if and only if  $u \in V(H_w^\tau)$  and set  $z_{\{u,v\}}^{w,\tau} = 1$  if and only if  $\{u, v\} \in E(H_w^\tau)$ . All other variables are 0. In particular, for  $\tau \notin L'$ , even  $z_\emptyset^{w,\tau} = 0$ .

Let us now prove that all the constraints are satisfied. Constraint (20) is obviously satisfied, since for any  $u \in V$  and  $\tau \in L^\lambda$  the variables in the *smES* LP are either all 0 or are integer variables corresponding to  $\tau$ -nearly regular subgraph. Constraint (21) is satisfied since  $|L'_u| \leq O(\log^3 \Delta)$ , and if  $\tau \notin L'_u$  then  $z_\emptyset^{u,\tau} = 0$ . Similarly, Constraint (22) is satisfied since  $|L'_u|$  and  $|L'_v|$  are both at most  $O(\log^3 \Delta)$  (and thus the left hand side is at most  $O(\log^3 \Delta)$ ), and if any one of the variables on the left is 1 then the edge  $\{u, v\}$  is present in  $E_H$  and thus  $x_{\{u,v\}} = 1$ . Finally, Constraint (24) is satisfied because for every edge  $\{u, v\} \in \hat{E}$ , if  $\{u, v\} \in E_H$  then  $x_{\{u,v\}} = 1$  and  $\{u, v\}$  does not appear as an edge in any *smES* instance, so the left hand side of the constraint is 1. Otherwise  $x_{\{u,v\}} = 0$ , in which case  $\{u, v\} \in H_{w(u,v)}$  but is not in any other *smES* instance. Since we decompose  $H_{w(u,v)}$ , there is exactly one  $\tau$  that covers  $\{u, v\}$  and for which the corresponding  $z_{\{u,v\}}^{w,\tau} = 1$ , and thus the left hand side is again 1.  $\square$



## 4 Reduction of LD2S to faithful roundings of $SmES$

We now show that if we are given an algorithm  $\mathcal{A}$  that is a factor  $f$  faithful rounding for  $\mathbf{SmES-LP}_q^\tau(G)$  (even restricted to the case when  $z_\emptyset = 1$ ), there is an  $\tilde{O}(f(\Delta))$ -approximation algorithm for LD2S that uses algorithm  $\mathcal{A}$  as a black box. Lemma 3.3 then implies that it is sufficient to be faithful for  $\mathbf{Bipartite-SmES-LP}_q^\tau(B(G))$ . Our reduction is given as Algorithm 2, which is relatively simple. It begins with all edges as the demand set  $\hat{E}$ , and first solves the LP relaxation for LD2S with demand set  $\hat{E}$ . It then adds every edge that has  $x$  value at least  $1/4$ . Then for every  $SmES$  instance in the relaxation it flips a coin, and with probability proportional to  $z_\emptyset$  (for that instance) scales all  $y$  variables for that instance by a factor of  $1/z_\emptyset$  and then uses the  $SmES$  algorithm on the instance. After this is completed we update the demands  $\hat{E}$  by removing edges that were successfully covered by this process, and repeat.

---

**Algorithm 2:** Approximation algorithm for LD2S

---

**Input** : Graph  $G = (V, E)$ , degree bound  $\lambda$ , factor  $f$  faithful rounding algorithm  $\mathcal{A}$  for  $\mathbf{SmES-LP}_q(\cdot)$

**Output:** 2-spanner  $H = (V, E_H)$  of  $G$  with maximum degree  $\lambda$

- 1 Let  $\hat{E} \leftarrow E$  (unsatisfied demand edges)
- 2 Let  $E_H \leftarrow \emptyset$  (spanner edges)
- 3 **while**  $\hat{E} \neq \emptyset$  **do**
- 4     Compute a solution  $\langle \vec{x}, \vec{z} \rangle$  for LP  $\mathbf{LD2S}_q^\lambda(G, \hat{E})$  from Figure 3
- 5      $E_x \leftarrow \{e \in E : x_e \geq 1/4\}$
- 6     **foreach**  $u \in V, \tau \in L^\lambda$  **do**
- 7         **with probability**  $z_\emptyset^{u,\tau}$
- 8              $H_{u,\tau} \leftarrow \mathcal{A}(G_u, \{z_b^{u,\tau}/z_\emptyset^{u,\tau}\}_{b \in V(G_u) \cup E(G_u)})$ ;     // output of  $SmES$  rounding
- 9              $E_{u,\tau} \leftarrow \{\{u, v\} \in E : v \in V(H_{u,\tau})\}$
- 10         **else (with remaining probability)**
- 11              $E_{u,\tau} \leftarrow \emptyset$
- // Add all edges found in above rounding
- 12      $E_H \leftarrow E_H \cup E_x \cup (\bigcup_{u \in V} \bigcup_{\tau \in L^\lambda} E_{u,\tau})$
- // Remove satisfied demands
- 13      $\hat{E} \leftarrow \hat{E} \setminus (E_H \cup \{\{u, v\} : \exists w \in V \text{ s.t. both } \{u, w\}, \{w, v\} \in E_H\})$

---

**Theorem 4.1.** *Let algorithm  $\mathcal{A}$  be a factor  $f$  faithful rounding for  $\mathbf{SmES-LP}_q^\tau(G)$  with  $z_\emptyset = 1$ . Then there is a (randomized)  $\tilde{O}(f(\Delta))$ -approximation for LD2S.*

*Proof.* We may assume that our LD2S algorithm has a valid guess for  $\lambda = \text{OPT}$  by simply trying all  $\Delta$  relevant values and reporting the best solution. In this case, the linear program  $\mathbf{LD2S}_q^\lambda(G, \hat{E})$  is guaranteed to have a feasible solution for any  $\hat{E} \subseteq E$ . We now use Algorithm 2 with this value of  $\lambda$ . The proof has two parts: first, we show that in each iteration of the main loop, with high probability the set of edges added to  $E_H$  (namely  $E_x \cup (\bigcup_{u \in V} \bigcup_{\tau \in L^\lambda} E_{u,\tau})$ ) forms a subgraph with maximum degree at most  $\tilde{O}(f(\Delta)) \cdot \lambda$ . Second, we show that with high probability there are only  $\tilde{O}(1)$  iterations of the main loop. Clearly these together prove the theorem.

We begin by analyzing the cost (i.e. maximum degree) of a single iteration of the main loop. Let  $u \in V$  be an arbitrary vertex; we will show that at most  $\tilde{O}(f(\Delta)) \cdot \text{OPT}$  edges were added incident to it. There are three types of edges incident to  $u$  that are added: these in  $E_x$ , those in  $E_{u,\tau}$  for some  $\tau$ , and those in  $E_{v,\tau}$  for some neighbor  $v$  of  $u$  and  $\tau \in L^\lambda$ . For the first type, the number of edges in  $E_x$  incident on  $u$  is at most  $|\{v \in \Gamma(u) : x_{\{u,v\}} \geq 1/4\}| \leq \sum_{v \in \Gamma(u)} 4x_{\{u,v\}} \leq 4\lambda$ , where we used Constraint (23) to bound the sum.

For the second type of edges, let  $L_u$  be the multiset of parameters from  $L^\lambda$  on which we actually used algorithm  $\mathcal{A}$  on  $u, \tau$  (i.e. lines 7 and 8 were executed). Then by Constraint (21) the expected size of  $L_u$  is at most  $\sum_{\tau \in L^\lambda} z_\emptyset^{u,\tau} \leq O(\log^3 n)$ , and since the random coins of algorithm  $\mathcal{A}$  are independent for each  $\tau$ , a simple Chernoff bound implies that this holds with high probability (say, probability more than  $1 - 1/n^4$ ). For each  $\tau \in L_u$ , the size of  $E_{u,\tau}$  is equal to  $|V(H_{u,\tau})|$ . We know from part 3 of the definition of faithful rounding that the size of this set is at most  $f(\Delta) \cdot \sum_{v \in \Gamma(u)} z_v^{u,\tau}$ , and now Constraint (13) from the *smES* LP (with  $T = \emptyset$ ) together with (18) implies that this is at most  $f(\Delta)(k_0 + k_1) \leq 2\lambda f(\Delta)$ . Thus the size of  $\bigcup_{\tau \in L^\lambda} E_{u,\tau}$  is with high probability at most  $O(\log^3 n)\lambda f(\Delta) \leq \tilde{O}(f(\Delta)) \cdot \text{OPT}$ .

For the third type of edges, let  $v \in \Gamma(u)$  and  $\tau \in L^\lambda$ . Then the probability that  $\{u, v\} \in E_{v,\tau}$  is at most  $z_\emptyset^{v,\tau}(1/z_\emptyset^{v,\tau})z_u^{v,\tau} = z_u^{v,\tau}$ , where the first  $z_\emptyset^{v,\tau}$  factor is from the probability of applying algorithm  $\mathcal{A}$  to this *smES* instance, the  $(1/z_\emptyset^{v,\tau})$  factor is from the scaling of the variables, and the  $z_u^{v,\tau}$  factor is from the definition of faithful rounding. Now (22) and a simple union bound imply that the probability that  $\{u, v\} \in \bigcup_{\tau \in L^\lambda} E_{v,\tau}$  is at most  $\sum_{\tau \in L^\lambda} z_u^{v,\tau} \leq O(\log^3 \Delta) \cdot x_{\{u,v\}}$ . This is independent for each  $v \in \Gamma(u)$ , so by a Chernoff bound we get that with high probability the number of type 3 edges is at most  $O(\log^3 n) \sum_{v \in \Gamma(u)} x_{\{u,v\}} \leq O(\log^3 n) \cdot \lambda \leq \tilde{O}(1) \cdot \text{OPT}$ , where we used Constraint (23) to bound the sum.

Now it just remains to show that only  $\tilde{O}(1)$  iterations of the main loop are necessary. We will show that in every iteration at least a  $c' = \tilde{\Omega}(1)$  fraction of the remaining demands  $\hat{E}$  are satisfied in expectation, or equivalently that in expectation the number of remaining unsatisfied demands is at most a  $(1 - c')$  fraction of the previous number of demands. To see that this is sufficient, note that by Markov's inequality with probability at most  $1 - c'/2$  the number of remaining demands is at least a  $\frac{1}{1-c'/2}(1 - c') = 1 - \frac{c'/2}{1-c'/2}$  fraction of what it was. Equivalently, with probability at least  $c'/2$  at least a  $\frac{c'/2}{1-c'/2}$  fraction of demands are covered. Thus the probability that this does not happen after  $(8/c') \ln n = \tilde{O}(1)$  iterations is at most  $(1 - c'/2)^{(8/c') \ln n} \leq 1/n^4$ . So with high probability, after  $\tilde{O}(1)$  rounds the number of unsatisfied demands is at most  $1 - \frac{c'/2}{1-c'/2} \leq 1 - c'/2$  of what it was. Now if this happens  $(2/c') \ln n = \tilde{O}(1)$  the number of remaining demands is at most  $|E|(1 - c')^{(2/c') \ln n} < 1$ , so the algorithm terminates. Thus with high probability the number of iterations is at most  $(8/c') \ln n \cdot (2/c') \ln n = \tilde{O}(1)$ , as required.

So now we just need to bound the expected number of demands satisfied in a single iteration, and show that this is  $\tilde{\Omega}(|\hat{E}|)$ . We break this into two cases. If  $\sum_{\{u,v\} \in \hat{E}} x_{\{u,v\}} \geq |\hat{E}|/2$ , then a simple averaging argument shows that at least a  $1/3$  fraction of the edges  $\{u, v\} \in \hat{E}$  are included in  $E_x$ , and since each one clearly covers itself as a demand a total of  $\Omega(|\hat{E}|)$  demands are covered.

In the second case we have that  $\sum_{\{u,v\} \in \hat{E}} x_{\{u,v\}} < |\hat{E}|/2$ . We can sum Constraint (24) over all demands in  $\hat{E}$ , giving us

$$\sum_{\{u,v\} \in \hat{E}} \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} z_{\{u,v\}}^{w,\tau} = |\hat{E}| - \sum_{\{u,v\} \in \hat{E}} x_{\{u,v\}} > |\hat{E}|/2 \quad (26)$$

For each  $\{u, v\} \in \hat{E}$  and  $w \in \Gamma(u) \cap \Gamma(v)$  and  $\tau \in L^\lambda$ , let  $p_{\{u,v\}}^{w,\tau}$  be the probability that the *smES* rounding of  $\mathbf{SmES-LP}_q^\tau(G_w)$  covers  $\{u, v\}$ . Note that  $p_{\{u,v\}}^{w,\tau}$  is just  $z_\emptyset^{w,\tau}$  times the probability that  $\{u, v\}$  is covered by the *smES* rounding of  $\mathbf{SmES-LP}_q^\tau(G_w)$  assuming that we actually perform this rounding. The expected total number of times edges in  $\hat{E}$  are covered in this phase (with repetitions), can be written as follows:

$$\begin{aligned}
\sum_{\{u,v\} \in \hat{E}} \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} p_{\{u,v\}}^{w,\tau} &= \sum_{w \in V} \sum_{\tau \in L^\lambda} \sum_{\{u,v\} \in \hat{E}: u,v \in \Gamma(w)} p_{\{u,v\}}^{w,\tau} \\
&\geq \sum_{w \in V} \sum_{\tau \in L^\lambda} z_\emptyset^{w,\tau} \cdot \tilde{\Omega} \left( \sum_{\{u,v\} \in \hat{E}: u,v \in \Gamma(w)} z_{\{u,v\}}^{w,\tau} / z_\emptyset^{w,\tau} \right) \\
&= \tilde{\Omega} \left( \sum_{\{u,v\} \in \hat{E}} \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} z_{\{u,v\}}^{w,\tau} \right) \\
&\geq \tilde{\Omega}(|\hat{E}|), \tag{by (26)}
\end{aligned}$$

where the first inequality is from the definition of  $p_{\{u,v\}}^{w,\tau}$  and part 4 of Definition 2.1.

Furthermore, we know from the second part of the definition of faithful rounding that  $p_{\{u,v\}}^{w,\tau} \leq z_\emptyset^{w,\tau} (z_{\{u,v\}}^{w,\tau} / z_\emptyset^{w,\tau}) = z_{\{u,v\}}^{w,\tau}$ , so by (24) we have

$$\sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} p_{\{u,v\}}^{w,\tau} \leq \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} z_{\{u,v\}}^{w,\tau} \leq 1.$$

We can then deduce that the probability that we cover demand  $\{u, v\} \in \hat{E}$  (in a single iteration) is at least  $\frac{1}{2} \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} p_{\{u,v\}}^{w,\tau}$ , by simply using the following well-known argument: if  $t$  (pairwise) independent events occur with probabilities  $q_1, \dots, q_t$  that sum up to  $\sum_{i=1}^t q_i \leq 1$ , then by Bonferroni inequality, the probability that at least one of these events occurs is at least

$$\sum_i q_i - \sum_{i < j} q_i q_j = \sum_i q_i - \frac{1}{2} \sum_{i \neq j} q_i q_j \geq \frac{1}{2} \sum_i q_i. \tag{27}$$

We thus obtain that the expected number of demands covered (in a single iteration) is at least

$$\frac{1}{2} \sum_{\{u,v\} \in \hat{E}} \sum_{w \in \Gamma(u) \cap \Gamma(v)} \sum_{\tau \in L^\lambda} p_{\{u,v\}}^{w,\tau} \geq \tilde{\Omega}(|\hat{E}|),$$

which completes the proof.  $\square$

## 5 Structure of faithful rounding algorithm for *smES*

We devote the remaining sections to our rounding algorithm for *smES*. Given an  $n$ -vertex bipartite graph  $G = (V_0, V_1, E)$ , and a solution to the LP relaxation<sup>6</sup>  $\mathbf{Bipartite-SmES-LP}_q(G, k_0, k_1, d_0, d_1)$ , our algorithm will give a factor- $f$  faithful rounding (see Definition 2.1), for some factor  $f = f(n, k_0, k_1, d_0, q)$  which we define in Section 6.1. Given the parameter  $q$  which determines the

<sup>6</sup> We will assume in the remaining sections that the LP solution is normalized. That is, we assume that  $y_\emptyset = 1$  (since otherwise, we normalize all variables by defining  $y'_T = y_T / y_\emptyset$ ).

size of our LP and hence the running time (as noted earlier, these are bounded by  $n^{O(q)}$ ), our approximation factor  $f$  will be at most  $\tilde{O}(n^{3-2\sqrt{2}+O(1/q)})$ .

At a high level, our algorithm finds a carefully chosen collection of (not necessarily induced) constant-size subgraphs of  $G$  in the form of caterpillars, and then samples vertices and edges from the union of these caterpillars according to a very specific distribution.<sup>7</sup>

Finally, in Appendix A, we give a much simpler faithful rounding algorithm which achieves approximation factor  $d_0$ . If it happens that  $d_0 \leq f$ , we will run the algorithm described in the appendix, and otherwise we will run our main algorithm. Thus, in the remaining sections, we will assume throughout that  $f \leq d_0$ . Recall that by convention we assume that  $k_0 \geq k_1$  and thus  $d_1 \geq \Omega(d_0/\log n)$ , so  $f \leq d_0 \leq d_1 \log n$ .

## 5.1 A simplified goal

Let us identify some simplified conditions which are sufficient in order to achieve a factor  $f$  faithful rounding. We start by weakening the definition of faithful rounding:

**Definition 5.1.** A randomized rounding algorithm  $\mathcal{A}$  is a *factor- $f$  weakly faithful rounding* if there is some deterministically chosen  $\varphi \leq 1$  (possibly  $o(1)$ ), such that when given a solution  $(y_a)_{a \in V, E}$  to an LP relaxation for  $smES$  on a graph  $G = (V, E)$ , the algorithm produces a random (not necessarily induced) non-empty subgraph  $H^* = (V^*, E^*)$  such that

1.  $\Pr[v \in V^*] \leq \varphi f \cdot y_v$  for all  $v \in V$ ,
2.  $\Pr[\{u, v\} \in E^*] \leq \varphi y_{\{u, v\}}$  for all  $\{u, v\} \in E$ ,
3.  $|V^*| \leq \varphi f \cdot \sum_{v \in V} y_v$  (with probability 1), and
4.  $\mathbb{E}[|E^*|] \geq \tilde{\Omega}\left(\varphi \cdot \sum_{\{u, v\} \in E} y_{\{u, v\}}\right)$ .

It turns out that this is equivalent to our original notion:

**Lemma 5.2.** *For every graph  $G$  and LP solution  $(y_a)$  as above, if there is an algorithm that achieves a weakly-faithful factor- $f$  rounding, then there is also an algorithm that achieves a faithful factor- $\tilde{O}(f)$  rounding.*

*Proof.* Run the weakly-faithful algorithm  $1/\varphi$  times, and take the union of all subgraphs returned. A trivial union bound implies that this algorithm satisfies the first three parts of the definition of faithful with factor  $f$ . To see that it satisfies the fourth part (that the expected number of edges is at least  $\tilde{\Omega}\left(\sum_{\{u, v\} \in E} y_{\{u, v\}}\right)$ ), let  $p_{\{u, v\}}$  denote the probability that the edge  $\{u, v\}$  is included in a single iteration. Since  $p_{\{u, v\}} \leq \varphi y_{\{u, v\}} \leq \varphi$ , the probability that  $\{u, v\}$  is included in at least one iteration is  $1 - (1 - p_{\{u, v\}})^{1/\varphi} \geq (1 - e^{-1})p_{\{u, v\}}/\varphi$ . So by linearity of expectations the expected number of edges covered in the union is at least  $\frac{1-e^{-1}}{\varphi} \sum_{\{u, v\} \in E} p_{\{u, v\}} \geq \tilde{\Omega}\left(\sum_{\{u, v\} \in E} y_{\{u, v\}}\right)$  as required (where the inequality is from part 4 of the definition of weakly faithful).  $\square$

<sup>7</sup>As described previously, when used to approximate LD2S the sampled edges are used only in the analysis while the sampled vertices are used to buy spanner edges

In Section 7.1, we will show that our rounding algorithm is faithful, with approximation factor determined by the set cardinalities (of subgraph vertices, and of subgraph edges). In other words, we show that it suffices to prove an approximation guarantee in the usual sense, and faithfulness will follow directly. Anticipating this shift in focus, let us define an even weaker notion of faithfulness (which will not be sufficient in general) in terms of set cardinalities:

**Definition 5.3.** A randomized rounding algorithm  $\mathcal{A}$  is a *skewed proportional rounding with parameters*  $(k'_0, k'_1, m')$  if when given a solution  $(y_a)_{a \in V, E}$  to an LP relaxation for *smES* with parameters  $(k_0, k_1, d_0, d_1)$  on a bipartite graph  $G = (V_0, V_1, E)$ , the algorithm produces a random (not necessarily induced) subgraph  $H^* = (V_0^*, V_1^*, E^*)$  such that

1.  $\Pr[u \in V_0^*] \leq \frac{k'_0}{k_0} \cdot y_u$  for all  $u \in V_0$ ,
2.  $\Pr[v \in V_1^*] \leq \frac{k'_1}{k_1} \cdot y_v$  for all  $v \in V_1$ ,
3.  $\Pr[e \in E^*] \leq \frac{m'}{m} \cdot y_e$  for all  $e \in E$ ,
4.  $|V_0^*| \leq k'_0$  and  $|V_1^*| \leq k'_1$  (with probability 1), and
5.  $\mathbb{E}[|E^*|] \geq \tilde{\Omega}(m')$ .

As defined, a skewed proportional rounding does not give us any guarantee. The reason for this is that the inflation factors relative to the LP values can be different for nodes in  $V_0$  and  $V_1$ . Namely, we might have  $\frac{k'_0}{k_0} \neq \frac{k'_1}{k_1}$ . Let us determine some sufficient condition on the parameters  $k'_0, k'_1, m'$  which will guarantee a faithful rounding. Suppose initially, we are given an algorithm which gives a skewed proportional rounding where  $k'_0 \gg k_0 f$  and  $k'_1 \gg k_1 f$ . We could prune the subgraph, by taking a uniformly chosen subset  $V'_0 \subseteq V_0^*$  of size  $(k_0 f / k'_0) |V_0^*|$ , and a uniformly chosen subset  $V'_1 \subseteq V_1^*$  of size  $(k_1 f / k'_1) |V_1^*|$ . The expected number of edges in the remaining subgraph (with vertices  $(V'_0, V'_1)$ ) is

$$m' \cdot \frac{k_0 f}{k'_0} \cdot \frac{k_1 f}{k'_1}.$$

If this is at least  $\tilde{\Omega}(m)$ , then it is easy to see that we have a faithful rounding. However, in the general case, this last condition may not be sufficient, since we might have  $k'_0 \ll k_0 f$  or  $k'_1 \ll k_1 f$  (or both), and then the pruning step does not apply. However, if we also have sufficient average degree in  $H^*$  (without pruning), that is,  $m'/k'_0 \geq d_0/f$  and  $m'/k'_1 \geq d_1/f$ , then this is sufficient.

**Lemma 5.4.** *Suppose we have a skewed proportional rounding with parameters  $(k'_0, k'_1, m')$  which satisfy the following three conditions:*

1.  $m'/k'_0 \geq d_0/f$ ,
2.  $m'/k'_1 \geq d_1/f$ ,
3.  $m' \cdot \frac{k_0 f}{k'_0} \cdot \frac{k_1 f}{k'_1} \geq m = \tilde{\Theta}(k_0 d_0)$ .

*Then we can also get a faithful factor- $f$  rounding.*

*Proof.* Let us consider two cases.

Case 1:  $k'_0 \geq k_0 f$  and  $k'_1 \geq k_1 f$ . In this case, as explained above, we can sample random subsets of  $V_0^*$  and  $V_1^*$  of the correct size in order to get the correct number of vertices on each side (Part 3 of Definition 2.1), and then by condition (3), the number of edges is sufficient (Part 4 of Definition 2.1). Note also, that the individual vertex probabilities are appropriately rescaled (giving Part 1 of Definition 2.1) by the sampling. Finally, let  $\rho = \frac{k_0 f}{k'_0} \cdot \frac{k_1 f}{k'_1}$  be the subsampling probability of edges in  $E^*$  (i.e. the probability that an edge in  $E^*$  is retained). If  $m' \rho = m$ , this implies both Parts 2 and 4 of Definition 2.1. Otherwise, subsample every remaining edge with probability  $m/(m' \rho)$ , which then guarantees both parts.

Case 2:  $k'_0 \leq k_0 f$  or  $k'_1 \leq k_1 f$ . Without loss of generality, suppose  $k'_0/k_0 \leq k'_1/k_1$ . In this case, let  $V'_0 = V_0^*$ , and take a uniformly chosen subset  $V'_1 \subseteq V_1^*$  of size  $(k'_0 k_1 / (k_0 k'_1)) |V_1^*|$ . Thus the vertex sets have the correct cardinalities for weakly faithful rounding (with scaling factor  $\varphi = k'_0 / (k_0 f)$ ). Since we only pruned on one side ( $V_1^*$ ), the expected average degree in  $V_1^*$  has not changed, which is all we need to show. More formally, the expected number of edges is at least  $m' k'_0 k_1 / (k_0 k'_1) \geq d_1 k'_0 k_1 / (f k_0) = m k'_0 / (f k_0) = \varphi \cdot m$ , where the inequality follows from condition (2). Parts 1 and 2 of Definition 5.1 follow by similar arguments to Case 1.  $\square$

**Corollary 5.5.** *Given a skewed proportional rounding with parameters  $(k'_0, k'_1, m')$ , then denoting by  $d'_b = m'/k'_b$  the average degree in  $S_b^*$  (for  $b = 0, 1$ ), the following three conditions suffice for a faithful rounding:*

1.  $d'_0 \geq d_0/f$ ,
2.  $d'_1 \geq d_1/f$ ,
3.  $\frac{d'_b}{k'_{1-b}} \geq \frac{d_0}{k_1 f^2}$  (for either  $b = 0$  or  $b = 1$ ).

**Corollary 5.6.** *Given a skewed proportional rounding with parameters  $(k'_0, k'_1, m')$ , then denoting by  $d'_b = m'/k'_b$  the average degree in  $S_b^*$  (for  $b = 0, 1$ ), the following three conditions suffice for a faithful rounding:*

1.  $d'_0 \geq d_0/f$ ,
2.  $d'_1 \geq d_1/f$ ,
3.  $k'_b \leq k_b f$  (for either  $b = 0$  or  $b = 1$ ).

*Proof.* Follows by substituting the lower bound  $d_{1-b}/f$  for  $m'/k'_{1-b}$  (Condition 1 or 2) in the original condition 3, and using  $k_0 d_0 = \tilde{\Theta}(k_1 d_1)$ .  $\square$

## 5.2 Algorithm description

Our algorithm is a variation of the non-faithful DKS algorithm of [BCC<sup>+</sup>10], and as with their algorithm is fundamentally concerned with *caterpillar graphs*. A caterpillar is a tree consisting of a main path (the *backbone*), with various disjoint paths (*hairs*) attached by one of their endpoints to the backbone. We concentrate on caterpillars with hair-length 1 (single edges). Let us state the following definition from [BCC<sup>+</sup>10], which forms the basic template for the algorithm on graphs with maximum degree  $n^{r/s}$ :

**Definition 5.7.** An  $(r, s)$ -caterpillar is a tree constructed inductively as follows: Begin with a single vertex as the leftmost node in the backbone. For  $s$  steps, do the following: at step  $t$ , if the interval  $((t-1)r/s, tr/s)$  contains an integer, add a hair of length 1 to the rightmost vertex in the backbone; otherwise, add an edge to the backbone (increasing its length by 1).

The algorithm will follow along the above iterative construction of an  $(r, s)$ -caterpillar. At step  $t$  of the construction, we will consider a union of  $t$ -edge prefixes of  $(r, s)$ -caterpillars in the graph  $G$ . These will be chosen by an elaborate pruning process described in Section 6.2, which will ensure certain uniformity properties while maintaining a large LP weight associated with these caterpillars. As part of the pruning process, at step  $t$ , we will consider a certain bipartite subgraph  $G_t = (S_t, W_t, E_t)$ , whose edges will come from the union of edges added in step  $t$  to the above caterpillars (the  $t$ 'th edge in the construction of each).

Furthermore, we will consider the tuple of leaves added before the  $t$ 'th edge in an  $(r, s)$ -caterpillar. Our bucketing will isolate a set of such leaf tuples  $\tilde{L}_t$ , and for every leaf tuple  $\lambda \in \tilde{L}_t$ , we will associate a certain subgraph  $H_t^\lambda$  of  $G_t$ , whose edges come from the  $t$ 'th edges in caterpillar prefixes whose initial leaves (up to the  $t$ th edge) correspond to  $\lambda$ . We will write the bipartite graph  $H_t^\lambda$  as  $H_t^\lambda = (U(H_t^\lambda), W(H_t^\lambda), E(H_t^\lambda))$ . Note that since we only consider constant size objects, the set  $\tilde{L}_t$  contains at most a polynomial number of tuples.

We are now ready to present our algorithm, in Figure 4, which will give a skewed-proportional rounding satisfying the conditions of Lemma 5.4 (though under certain conditions the rounding may also be weakly faithful, or even faithful). We remind the reader that this algorithm is intended only for parameters satisfying  $f \leq d_0$  (otherwise, we use the much simpler algorithm in Appendix A). We defer the details of how the above sets are defined to Section 6.2. For now, we only note that they can be computed in parallel to the following algorithm, as needed. Our approximation guarantee  $f$  and parameters  $r, s$  (all of which depend on  $n, k_0, k_1, d_0, d_1$ , and  $q$ ) will be defined in Section 6.1.

## 6 Details of our rounding algorithm for bipartite $SmES$

In this section, we give details of our bucketing procedure which defines the various subgraphs used in algorithm **Faithful-SmES**. Moreover, we specify the parameters  $r, s$  and our approximation ratio  $f$  which define the specific caterpillar structure on which our algorithm is based.

### 6.1 Parametrization and maximum degree bound

Given parameters  $n, k_0, k_1, d_0, d_1, q$ , we need to define our intended approximation ratio  $f$ , and the choice of  $r$  and  $s$  in the  $(r, s)$ -caterpillar which will determine the structure of our graph, and will also give an effective bound on the maximum degree. In this section, we define these parameters. As before, let  $q$  to be the (bounded) integer parameter which corresponds to the level of the Sherali-Adams relaxation we use. The running time both for solving the LP and for our rounding will be  $n^{O(q)}$  while the approximation guarantee will be  $\tilde{O}(n^{3-2\sqrt{2}+O(1/q)})$ .

Similarly to [BCC<sup>+</sup>10], we would like to choose  $r$  and  $s$  so that the maximum degree will be roughly  $n^\alpha = n^{r/s}$  for  $s \leq q$ . In Appendix B, we will show that Step 1 of the algorithm gives a faithful rounding whenever the maximum degree is greater than  $D = \frac{n}{k_1 f} \cdot \frac{d_0}{f}$ . Thus we would to define  $\alpha = r/s$  so that

$$n^\alpha \approx \frac{n}{k_1 f} \cdot \frac{d_0}{f}. \quad (28)$$

**Faithful-SmES**( $G, \{y_I\}$ )

Input: A graph  $G$  with LP solution  $\{y_I\}$  to **Bipartite-SmES-LP** $_q(G, k_0, k_1, d_0, d_1)$ .

For steps  $t = 1, \dots, s$ :

1. If the maximum degree in  $G_t$  is at least  $\tilde{\Omega}\left(\frac{nd_0}{k_1 f^2}\right)$ , do the following, and halt:
  - If  $S_t \subseteq V_0$ , choose a subset of  $k_0 f$  vertices in  $S_t$  uniformly at random, and choose  $k_1 f$  vertices in  $W_t$  uniformly at random. Otherwise, choose  $k_1 f$  vertices in  $S_t$  and  $k_0 f$  in  $W_t$  uniformly at random. Output these vertices.
  - For every edge  $e = (u, v)$  whose endpoints are chosen in the previous step, choose edge  $e$  with probability  $\frac{y_{\{e\}}}{(y_{\{u\}}y_{\{v\}})^2}$ . Output these edges.
2. Let  $m' = \mathbb{E}_{\lambda \in_R \tilde{L}_t}[|E(H_t^\lambda)|]$ . If  $S_t \subseteq V_0$ , let  $k'_0 = \mathbb{E}_{\lambda \in_R \tilde{L}_t}[|U(H_t^\lambda)|]$  and  $k'_1 = \mathbb{E}_{\lambda \in_R \tilde{L}_t}[|W(H_t^\lambda)|]$ . Otherwise, let  $k'_0 = \mathbb{E}_{\lambda \in_R \tilde{L}_t}[|W(H_t^\lambda)|]$  and  $k'_1 = \mathbb{E}_{\lambda \in_R \tilde{L}_t}[|U(H_t^\lambda)|]$ .
3. If the parameters  $m', k'_0, k'_1$  satisfy the conditions of Lemma 5.4, choose  $\lambda \in \tilde{L}_t$  uniformly at random, output  $H_t^\lambda$ , and halt.

Figure 4: Algorithm **Faithful-SmES**

We will also need our approximation factor  $f = f(n, k_0, k_1, d_0, d_1, q)$  to satisfy the following crucial property:

$$f = \left(\frac{k_1 f}{d_0}\right)^\alpha. \quad (29)$$

Note that if we had equality in both (28) and (29), then together, these implicitly define both  $\alpha$  and  $f$ . However, recall that we require  $\alpha = r/s$  to be rational (specifically, we will want  $\alpha$  to be a multiple of  $1/q$ ), which most likely would not occur if we had strict equality in (28). To fix this, we simply round the implied value of  $\alpha$  up to the nearest multiple of  $1/q$ . It is straightforward to check that, if we let  $\gamma = \log_n(k_1/d_0)$ , then the new value of  $\alpha$  will be

$$\alpha = \frac{1}{q} \left\lceil q(1 + \gamma/2 - \sqrt{2\gamma + \gamma^2/4}) \right\rceil.$$

This specific closed-form expression for  $\alpha$  is never used in our analysis. It is only important in that it guarantees (28), implicitly defines  $f$  through (29), and gives the following guarantee, which is the precise formulation of (28):

**Proposition 6.1.** *For  $f$ ,  $D$  and  $\alpha$  defined as above, we have*

$$D = \frac{n}{k_1 f} \cdot \frac{d_0}{f} = n^{\alpha - O(1/q)}.$$

This implies that

$$\begin{aligned} f &= n^{1-\alpha+O(1/q)} d_0 / (k_1 f) \\ &= n^{1-\alpha+O(1/q)} / f^{1/\alpha}. \end{aligned} \quad \text{by (29)}$$

Thus, in the worst case, we have the following upper bound on our approximation factor  $f$ :



**Corollary 6.2.** For  $f$ ,  $D$  and  $\alpha$  defined as above,

$$f = n^{(1-\alpha+O(1/q))\alpha/(1+\alpha)} \leq \max_{0 \leq \alpha \leq 1} n^{(1-\alpha)\alpha/(1+\alpha)+O(1/q)} = n^{3-2\sqrt{2}+O(1/q)} \approx n^{0.172+O(1/q)}.$$

Finally, we choose integers  $r, s$  such that  $r$  and  $s$  are co-prime and  $r/s = \alpha$ .

## 6.2 LP rounding and bucketing

The fundamental difficulty in adapting LP-hierarchy-based algorithms such as the one in [BCC<sup>+</sup>10] to work in a faithful randomized rounding setting is that the LP values which are finally used in finding the actual subgraph are values which arise after several rounds of conditioning. These can be markedly different from (or even nearly unrelated to) the original LP values, which we want to round with respect to. We get around this problem by limiting our algorithm to vertices (and edges, and larger structures) which have roughly uniform LP values (for every specific kind of vertex/edge/structure). Thus, even if the values which we use to round are different from the original LP values, they are at least uniformly proportional to them.

In ignoring many vertices/edges/structures in the graph, we have to make sure that we are not losing too much of the information given to us by the LP. Since our algorithm and analysis ultimately rely on examining caterpillars of a certain form (which is determined by our parameters), the goal of our bucketing of caterpillars is to preserve the total LP weight of such caterpillars in our graph, up to a polylogarithmic factor.

Let  $K$  be the template  $(r, s)$  caterpillar for  $r, s$  corresponding to our parameters (see Definition 5.7) and denote by  $K_t$  the  $(t-1)$ -edge  $(t)$ -vertex prefix of this caterpillar (that is, the caterpillar constructed inductively as before, for the first  $t-1$  steps). We will now define a procedure which buckets instances in  $G$  of these caterpillar prefixes.

Before describing the procedure, let us introduce one more piece of notation. For a set of caterpillars  $B$  (a bucket), for every tuple of leaves  $\lambda$  (not including the rightmost backbone vertex, which may be also be a leaf), denote by  $T^\lambda(B)$  the set of caterpillars in  $B$  which have leaves  $\lambda$  (in the same order). In addition, for node  $u$ , and edge  $e$ , denote by  $T_u^\lambda(B)$  and  $T_e^\lambda$  the set of caterpillars in  $T^\lambda(B)$  which have  $u$  (resp.  $e$ ) as the final vertex (resp. edge). Note that this edge may be either a backbone edge or a hair. An important invariant will be that at the end of every step, the cardinalities of sets  $|T^\lambda(B)|$  (resp.  $|T_u^\lambda(B)|$ ) will be roughly uniform (up to polylogarithmic factors) for every remaining choice of leaf tuple  $\lambda$  (resp. leaf tuple  $\lambda$  and rightmost backbone vertex  $u$ ).

- Initialization: By Constraint (13),  $\sum_{v \in V_0} y_v = k_0$ . Bucket vertices in  $V_0$  by their LP-value. Then there is some bucket  $B_1$  for which  $\sum_{v \in V_0} y_v \geq k_0 / \log n$ . Let us also write  $L_1 = S_1 = B_1$ .
- Step  $t$ : Suppose  $B_t$  is a bucket of instances of caterpillar  $K_t$  in the graph,  $S_t$  is the set of rightmost (final) backbone nodes in these caterpillars, and  $L_t$  is the set of leaf-tuples occurring in these caterpillars (not including the final backbone vertex). Let  $b \in \{0, 1\}$  be such that  $S_t \subseteq V_b$  (this depends only on the parity of backbone edges in  $K_t$ ). Construct a sequence of nested sets of caterpillars in stages as follows:
  - Consider the set of caterpillars  $\tilde{B}_t$  formed by taking a caterpillar from  $B_t$  and adding an edge to the rightmost node. Then up to a logarithmic factor, by (14), for every caterpillar  $K \in B_t$  with rightmost endpoint  $u$ , we have  $\sum_{w \in \Gamma(u)} y_{K \cup \{u, w\}} = d_b y_K$ . In particular, this implies  $\sum_{\tilde{K} \in \tilde{B}_t} y_{\tilde{K}} = d_b \cdot \sum_{K \in B_t} y_K$ .

- Bucket the new caterpillars in  $\tilde{B}_t$  so that the following values are uniform (up to a constant factor): the LP value of the newly added vertex, of the newly added edge, of the leaves together with the new vertex (that is, if the caterpillar originally had leaves  $\lambda$  and a new vertex  $w$  was added, then this is the LP value  $y_{\lambda \cup \{w\}}$ ) and of the entire caterpillar. Then the heaviest bucket preserves the total LP weight of caterpillars in  $\tilde{B}_t$  up to a polylogarithmic factor. Denote this bucket by  $\tilde{B}'_t$ .
- Next, bucket the caterpillars in  $\tilde{B}'_t$  so that within every bucket, for every tuple of leaves  $\lambda$  and edge  $(u, w)$  the cardinality  $|T_{(u,w)}^\lambda(\tilde{B}'_t)|$  is roughly uniform. Again, retain the heaviest bucket (which preserves at least a polylogarithmic fraction of LP weight), and denote this bucket by  $\tilde{B}''_t$ .
- Now bucket  $\tilde{B}''_t$  by the cardinality  $|\bigcup_u T_{(u,w)}^\lambda(\tilde{B}''_t)|$  (so that this value is roughly uniform over the choice of leaf tuple  $\lambda$  and new vertex  $w$ ), and let  $\tilde{B}'''_t$  denote bucket with the largest LP weight.
- Finally, prune the remaining set of caterpillars  $\tilde{B}'''_t$  by retaining only those with leaf tuples  $\lambda$  which satisfy

$$\sum_{K''' \in T^\lambda(\tilde{B}'''_t)} y_{K'''} \geq \tilde{\Omega}(d_b) \sum_{K \in T^\lambda(B_t)} y_K.$$

Note that for such  $\lambda$ , at least a polylogarithmic fraction of the total LP weight of  $T^\lambda(\tilde{B}_t)$  must have also been preserved in each of the previous stages. Denote this set of leaf tuples by  $\tilde{L}_t$ , and let  $B_{t+1} \subseteq \tilde{B}'''_t$  be the corresponding set of caterpillars.

- Denote by  $W_t$  the set of all newly added vertices in caterpillars in  $B_{t+1}$ . That is,

$$W_t = \bigcup_{\lambda, u} \{w \mid T_{(u,w)}^\lambda(B_{t+1}) \neq \emptyset\}.$$

- Backbone step: If the last edge in  $K_{t+1}$  is a backbone edge, let  $L_{t+1} = \tilde{L}_t$ , and let  $S_{t+1} = W_t$ .
- Hair step: Otherwise, the last edge in  $K_{t+1}$  is a hair. In this case, the rightmost backbone vertices in the caterpillars in  $\tilde{B}_t$  still belong to  $S_t$  (in each caterpillar that survives the bucketing, it is the same vertex as before the new edge was added). Let  $L_{t+1}$  be the set of new leaf tuples of these caterpillars (where each leaf tuple includes the newly added leaf in  $W_t$ ), and let  $S_{t+1} \subseteq S_t$  be the remaining set of rightmost backbone vertices.

Note that (by induction), if caterpillar  $K_t$  rooted in  $V_0$  has  $t_0$  edges emanating from side  $V_0$  and  $t_1$  edges emanating from side  $V_1$  (think of edges as being directed away from the initial node), then bucket  $B_t$  satisfies

$$\sum_{K \in B_t} y_K = \tilde{\Theta}(k_0 d_0^{t_0} d_1^{t_1}), \quad (30)$$

which is precisely the number of such caterpillars in a  $(d_0, d_1)$ -regular bipartite graph (assuming we allow degeneracies, such as caterpillars intersecting themselves).

The following lemma shows that LP weight is preserved even on substructures of caterpillars, such as individual vertices and edges.

**Lemma 6.3.** *Let  $B$  be a set of trees rooted on side  $V_0$  and isomorphic to a tree  $Q$  which, when rooted on side  $V_0$  with edges directed away from the root, has  $t_0$  edges emanating from side  $V_0$  and  $t_1$  edges emanating from side  $V_1$ . If the set  $B$  satisfies Equation (30), then*

- *For any vertex  $i$  in  $Q$  on side  $V_b$  (for some  $b \in \{0, 1\}$ ), letting  $B_i$  be the set of all copies of vertex  $i$  in trees  $R \in B$ , we have*

$$\sum_{v \in B_i} y_{\{v\}} = \tilde{\Theta}(k_b).$$

- *For any edge  $e$  in  $Q$ , letting  $B_e$  be the set of all copies of vertex  $e$  in trees  $R \in B$ , we have*

$$\sum_{g \in B_e} y_{\{g\}} = \tilde{\Theta}(k_0 d_0).$$

*Proof.* We give the proof for vertices on side  $V_0$ . The proof for vertices on side  $V_1$  and for edges is similar. Let  $s = \sum_{v \in B_i} y_{\{v\}}$ . By repeated applications of the Sherali-Adams Constraint (14) we have

$$\sum_{R \in B} y_R \leq s d_0^{t_0} d_1^{t_1}.$$

By (30), this implies that  $s = \tilde{\Omega}(k_0)$ . On the other hand, by Constraint (13), we have  $s \leq k_0$ , which completes the proof.  $\square$

*Remark 6.4.* We will take advantage of the bucketing on LP values and other values above to abuse notation. For example, since all caterpillars of the form  $K_t$  that survive our bucketing will have the same LP value (up to a constant factor), we can be a bit imprecise, and write  $y_{K_t}$ , with the understanding that we ignore constant factors in our analysis.

At step  $t$ , for leaf tuple  $\lambda \in \tilde{L}_t$ , let us define the bipartite subgraph  $H_t^\lambda = (U(H_t^\lambda), W(H_t^\lambda), E(H_t^\lambda))$  as follows. Let

$$E(H_t^\lambda) = \{(u, w) \mid T_{(u,w)}^\lambda(B_{t+1}) \neq \emptyset\},$$

and let  $U(H_t^\lambda)$  and  $W(H_t^\lambda)$  be the vertex sets incident to these edges. That is,  $U(H_t^\lambda) = \{u \in S_t \mid \exists w \in W_t : (u, w) \in E(H_t^\lambda)\}$ , and  $W(H_t^\lambda) = \{w \in W_t \mid \exists u \in U(H_t^\lambda) : (u, w) \in E(H_t^\lambda)\}$ . This subgraph (for an appropriate choice of  $t$ , and distribution over  $\lambda$ ) will form the basis of our rounding algorithm. Later, we will show that the cardinalities of sets involved do not depend on the choice of leaves (see Lemma 7.2).

Note that the graphs  $H_t^\lambda$  are subgraphs of the graphs  $G_t = (S_t, W_t, E_t)$  which were mentioned in Section 5.2. The graph  $G_t$  is in fact the union (over leaf tuples  $\lambda$ ) of all subgraphs  $H_t^\lambda$ . The sets  $S_t$  and  $W_t$  were defined in the above bucketing, while  $E_t$  is defined as follows:

$$E_t = \{(u, w) \mid T_{(u,w)}(B_{t+1}) \neq \emptyset\}.$$

## 7 Performance guarantee of our rounding algorithm: faithfulness and approximation

In this section, we show that Algorithm **Faithful-SmES** gives a faithful  $f$ -approximation (using Lemma 5.4). In fact, if at any iteration the lower bound on degrees in Step 1 of the algorithm

is satisfied, then Step 1 already gives a faithful rounding on its own. Otherwise (assuming an upper bound on degrees), we show that at some iteration  $t$ , Step 3 of the algorithm gives a skewed-proportional rounding satisfying the conditions of Lemma 5.4.

## 7.1 LP rounding: faithfulness

Before we prove the approximation guarantee, we need to show that the algorithm gives a skewed-proportional rounding. In fact, this represents the core of our technical contribution. We will show that the sampling procedure suggested in Step 3 of the algorithm always gives a skewed-proportional rounding, regardless of whether the conditions of Lemma 5.4 are met. The following lemma allows us to reformulate the conditions of skewed-proportional rounding in terms of set cardinalities.

**Lemma 7.1.** *Let  $\mathcal{A}$  be an algorithm which, for some  $t$ , outputs a random subgraph  $H^* = (V_b^*, V_{1-b}^*, E^*)$  of  $G_t = (S_t, W_t, E_t)$  (where  $b \in \{0, 1\}$  is s.t.  $S_t \subseteq V_b$ ), where the cardinalities  $|V_0^*|, |V_1^*|, |E^*|$  are roughly uniform (over the randomness). If the following conditions hold:*

1. For all  $u \in S_t$ ,  $\Pr[u \in V_b^*] = \tilde{O}(\mathbb{E}[|V_b^*|]/|S_t|)$ ,
2. for all  $w \in W_t$ ,  $\Pr[w \in V_{1-b}^*] = \tilde{O}(\mathbb{E}[|V_{1-b}^*|]/|W_t|)$ , and
3. for all  $e \in E_t$ ,  $\Pr[e \in E^*] = \tilde{O}(\mathbb{E}[|E^*|]/|E_t|)$ ,

then algorithm  $\mathcal{A}$  is a skewed proportional rounding with parameters  $(\tilde{O}(\mathbb{E}[|V_0^*|]), \tilde{O}(\mathbb{E}[|V_1^*|]), \tilde{O}(\mathbb{E}[|E^*|]))$ .

*Proof.* Follows directly from the definition of skewed proportional rounding, and the fact that, by bucketing and by Lemma 6.3, for all  $u \in S_t$ ,  $w \in W_t$  and  $e \in E_t$  we have  $y_u = \tilde{\Theta}(k_b/|S_t|)$ ,  $y_w = \tilde{\Theta}(k_{1-b}/|W_t|)$ , and  $y_e = \tilde{\Theta}(m/|E_t|)$ .  $\square$

Our goal is thus to show three simple lemmas corresponding to the above conditions. One each for the probability of a vertex/edge belonging to  $U(H_t^\lambda)$ ,  $W(H_t^\lambda)$ , and  $E(H_t^\lambda)$  (for uniformly chosen  $\lambda \in \tilde{L}_t$ ). First, let us show that, indeed, the cardinalities of these sets (and others) do not vary by more than a polylogarithmic factor over the choice of leaf-tuple  $\lambda$ .

**Lemma 7.2.** *For every step  $t$  in the above bucketing, the cardinality of both of the following sets (when non-empty) does not vary by more than a polylogarithmic factor over the choice of leaf tuple  $\lambda \in L_t$  and rightmost vertex  $u$ :*

$$T^\lambda(B_t), \quad T_u^\lambda(B_t).$$

Moreover, the cardinality of each of the following sets also does not vary by more than a polylogarithmic factor over the choice of leaf tuple  $\lambda \in \tilde{L}_t$ :

$$U(H_t^\lambda), \quad W(H_t^\lambda), \quad E(H_t^\lambda).$$

*Proof.* We proceed by induction. For  $t = 1$ , by definition, for every  $v \in L_1$  we have  $T^v(B_1) = \{v\}$ , and  $T_u^v(B_1) = \{v\}$  iff  $v = u$  (otherwise it is empty).

Let us now assume that the lemma holds for  $T^\lambda(B_t)$  and  $T_u^\lambda(B_t)$  (over the choice of  $\lambda \in L_t$ ), and show that it holds for  $U(H_t^\lambda)$ ,  $W(H_t^\lambda)$ ,  $E(H_t^\lambda)$  (over the choice of  $\lambda \in \tilde{L}_t$ ), and for  $T^\lambda(B_{t+1})$  and  $T_u^{\lambda'}(B_{t+1})$  (over the choice of  $\lambda' \in L_{t+1}$ ). Let  $\lambda \in \tilde{L}_t$  be a tuple of leaves. As noted earlier (see definition of  $\tilde{L}_t$ ), for any  $\lambda \in \tilde{L}_t$ , at least a polylogarithmic fraction of the total LP weight of

$T^\lambda(\tilde{B}_t)$  must have been preserved in each stage of the bucketing. Since by the inductive hypothesis, the cardinality  $|T_u^\lambda(B_t)|$  is roughly uniform (up to a polylogarithmic factor), and since LP values of caterpillars in  $B_t$  are roughly uniform, every vertex  $u$  with non-empty  $T_u^\lambda(B_t)$  contributes roughly the same LP weight to the total weight of  $T^\lambda(B_t)$ , and therefore a  $\Omega(1)$  fraction of these vertices must also survive the bucketing. In particular, this implies that

$$|U(H_t^\lambda)| = \tilde{\Theta}(|\{u \in S_t \mid T_u^\lambda(B_t) \neq \emptyset\}|) = \tilde{\Theta}(|T^\lambda(B_t)|/|T_u^\lambda(B_t)|). \quad (31)$$

Since by the inductive hypothesis, the two values in the final ratio on do not depend on  $\lambda, u_0$  (by more than a polylogarithmic factor), the claim follows for  $U(H_t^\lambda)$ .

Next, let us show uniformity of the cardinalities  $W(H_t^\lambda)$  and  $E(H_t^\lambda)$ . By construction, we have

$$|T^\lambda(B_{t+1})| = \frac{1}{y_{K_{t+1}}} \sum_{\tilde{K} \in T^\lambda(B_{t+1})} y_{\tilde{K}} = \tilde{\Theta} \left( \frac{d_b}{y_{K_{t+1}}} \right) \sum_{K \in T^\lambda(B_t)} y_K = \tilde{\Theta} \left( \frac{d_b y_{K_t}}{y_{K_{t+1}}} \right) |T^\lambda(B_t)|.$$

Since by construction of  $\tilde{B}_t''$ , every edge  $(u, w) \in E(H_t^\lambda)$  participates in the same number of caterpillars in  $T^\lambda(B_{t+1})$  (up to a constant factor), call it  $|T_e^\lambda(B_{t+1})|$ , we have that

$$\begin{aligned} |E(H_t^\lambda)| &= \frac{|T^\lambda(B_{t+1})|}{|T_e^\lambda(B_{t+1})|} \\ &= \tilde{\Theta} \left( \frac{d_b y_K |T^\lambda(B_t)|}{y_{K_{t+1}} |T_e^\lambda(B_{t+1})|} \right), \end{aligned} \quad (32)$$

which does not depend on  $\lambda$ , since all the values in the final expression are fixed by bucketing. By a similar argument (by construction of  $\tilde{B}_t'''$ ), every vertex  $w \in W(H_t^\lambda)$  has the same degree in  $H_t^\lambda$  (up to a constant factor). Thus, since the number of edges in  $H_t^\lambda$  is fixed (up to a polylogarithmic factor), so is  $|W(H_t^\lambda)|$ .

Finally, we need to show the uniformity of  $T^{\lambda'}(B_{t+1})$  and  $T_{u'}^{\lambda'}(B_{t+1})$  (over the choice of  $\lambda' \in L_{t+1}$  and  $u'$  such that  $T_{u'}^{\lambda'}(B_{t+1}) \neq \emptyset$ ). Let us consider the two different kinds of steps separately. If  $t$  is a backbone step, then by the bucketing that defines  $\tilde{B}_t''''$ , the number of caterpillars  $|T_w^\lambda(B_{t+1})|$  is fixed up to a constant factor. Moreover,  $\{w \in S_{t+1} \mid T_w^\lambda(B_{t+1}) \neq \emptyset\} = W(H_t^\lambda)$ , and as we've shown, the number of vertices  $w$  in the latter set is fixed up to a polylogarithmic factor, therefore the same holds for the total number of caterpillars  $|T^\lambda(B_{t+1})| = \sum_{w \in W(H_t^\lambda)} |T_w^\lambda(B_{t+1})|$ .

Now, suppose  $t$  is a hair step. As we've noted, for every  $(u, w) \in E(H_t^\lambda)$ , the number of caterpillars  $|T_{(u,w)}^\lambda(B_{t+1})|$  is fixed up to a constant factor. However, recall that in a hair step we have  $T_{(u,w)}^\lambda(B_{t+1}) = T_u^{\lambda \cup \{w\}}(B_{t+1})$ . Thus, it only remains to show the claim for  $|T^{\lambda \cup \{w\}}(B_{t+1})| = \sum_{u \in \Gamma_{H_t^\lambda}(w)} |T_u^{\lambda \cup \{w\}}(B_{t+1})|$ . But here it is clearly sufficient to show that every vertex  $w \in W(H_t^\lambda)$  has roughly the same degree in  $H_t^\lambda$ , which we have already argued.  $\square$

Before we state and prove the three main lemmas which we need for the probability bounds in Lemma 7.1, let us introduce the following notation. Let us assume that when caterpillar  $K_t$  has its initial vertex at side  $V_0$  with edges directed away from the initial vertex, it has  $t_0$  outgoing edges from side  $V_0$  and  $t_1$  outgoing edges from side  $V_1$  (thus  $t_0 + t_1 = t - 1$ ). For concreteness, let us assume that  $K_t$  has an even number of backbone edges. That is, when the initial backbone vertex is in  $V_0$  so is the rightmost vertex. The other case (when the backbone has odd length) is quite similar. We are now ready to prove the remaining lemmas which will guarantee skewed proportionality.

**Lemma 7.3.** For uniformly chosen  $\lambda \in \tilde{L}_t$ , for every  $u \in S_t$  we have  $\Pr_\lambda[u \in U(H_t^\lambda)] = \tilde{O}(|U(H_t)|/|S_t|)$ .

*Proof.* As we have argued earlier, we have  $\sum_{\lambda \in \tilde{L}_t} \sum_{K \in T^\lambda(B_t)} y_K = \tilde{\Omega}(k_0 d_0^{t_0} d_1^{t_1})$ . By bucketing, we can write this more simply as

$$|\tilde{L}_t| |T^\lambda(B_t)| y_{K_t} = \tilde{\Omega}(k_0 d_0^{t_0} d_1^{t_1}). \quad (33)$$

Now, take any vertex  $u \in S_t$ . By repeated applications of Constraint (14), we have

$$\sum_{\lambda \in \tilde{L}_t: U(H_t^\lambda) \ni u} \sum_{K \in T_u^\lambda(B_t)} y_K \leq d_0^{t_0} d_1^{t_1} y_u.$$

By Lemma 7.2, for any  $\lambda \in \tilde{L}_t$  such that  $U(H_t^\lambda) \ni u$ , the number of caterpillars this tuple of leaves contributes,  $|T_u^\lambda(B_t)|$ , is roughly uniform (up to a polylogarithmic factor). Thus, we have

$$|\{\lambda \in \tilde{L}_t \mid U(H_t^\lambda) \ni u\}| |T_u^\lambda(B_t)| y_{K_t} \approx \sum_{\lambda \in \tilde{L}_t: U(H_t^\lambda) \ni u} \sum_{K \in T_u^\lambda(B_t)} y_K \leq d_0^{t_0} d_1^{t_1} y_u.$$

By Constraint (13) and bucketing, we have  $y_u = O(k_0/|S_t|)$ , so the above inequality implies

$$|\{\lambda \in \tilde{L}_t \mid U(H_t^\lambda) \ni u\}| |T_u^\lambda(B_t)| y_{K_t} = O(k_0 d_0^{t_0} d_1^{t_1} / |S_t|).$$

Combining this with (33), we have

$$\begin{aligned} \Pr_{\lambda \in_R \tilde{L}_t}[u \in U(H_t^\lambda)] &= \frac{|\{\lambda \in \tilde{L}_t \mid U(H_t^\lambda) \ni u\}|}{|\tilde{L}_t|} \\ &= \tilde{O} \left( \frac{k_0 d_0^{t_0} d_1^{t_1} / (|S_t| |T_u^\lambda(B_t)| y_{K_t})}{k_0 d_0^{t_0} d_1^{t_1} / (|T^\lambda(B_t)| y_{K_t})} \right) \\ &= \tilde{O} \left( \frac{|T^\lambda(B_t)| / |T_u^\lambda(B_t)|}{|S_t|} \right), \end{aligned}$$

which by (31) is what we needed to prove.  $\square$

**Lemma 7.4.** For uniformly chosen  $\lambda \in \tilde{L}_t$ , for every  $w \in W_t$  we have  $\Pr_\lambda[w \in W(H_t^\lambda)] = \tilde{O}(|W(H_t)|/|W_t|)$ .

*Proof.* The proof is similar to that of Lemma 7.3. For  $\lambda \in \tilde{L}_t$ , let us denote by  $T^{\lambda,w}(B_{t+1})$  the set of caterpillars in  $B_{t+1}$  which have leaves  $\lambda$  and  $w$ . That is, if  $t$  is a backbone step, then  $T^{\lambda,w}(B_{t+1}) = T_w^\lambda(B_{t+1})$ , and otherwise (if  $t$  is a hair step), then  $T^{\lambda,w}(B_{t+1}) = T^{\lambda \cup w}(B_{t+1})$ . Noting that in both cases, the cardinality of this set is roughly uniform for all  $\lambda \in \tilde{L}_t$  and  $w \in W(H_t^\lambda)$ , let us abuse notation and write  $|T^{\lambda,w}|$  for the cardinality of such sets. As argued earlier, the degrees of all  $w \in W(H_t^\lambda)$  in  $H_t^\lambda$  are roughly uniform, and each edge  $(u, v) \in E(H_t^\lambda)$  participates in roughly the same number of caterpillars. Thus, it follows that

$$|W(H_t)| = \tilde{\Theta}(|T^\lambda(B_{t+1})| / |T^{\lambda,w}(B_{t+1})|). \quad (34)$$

Analogously to (33), we can also show

$$|\tilde{L}_t| |T^\lambda(B_{t+1})| y_{K_{t+1}} = \tilde{\Omega}(k_0 d_0^{t_0+1} d_1^{t_1}). \quad (35)$$

Now, take any vertex  $w \in W_t$ . By repeated applications of Constraint (14), we have

$$\sum_{\lambda \in \tilde{L}_t: W(H_t^\lambda) \ni w} \sum_{K \in T^{\lambda, w}(B_t)} y_K \leq k_0 d_0^{t_0} d_1^{t_1+1}.$$

As before, for any  $\lambda \in \tilde{L}_t$  such that  $W(H_t^\lambda) \ni w$ , the number of caterpillars this tuple of leaves contributes,  $|T^{\lambda, w}(B_t)|$ , is roughly uniform (up to a polylogarithmic factor). Thus, we have

$$|\{\lambda \in \tilde{L}_t \mid W(H_t^\lambda) \ni w\}| |T^{\lambda, w}(B_{t+1})| y_{K_{t+1}} \approx \sum_{\lambda \in \tilde{L}_t: W(H_t^\lambda) \ni w} \sum_{K \in T^{\lambda, w}(B_{t+1})} y_K \leq d_0^{t_0} d_1^{t_1+1} y_w.$$

By Constraint (13) and bucketing, we have  $y_w = O(k_1/|W_t|)$ , so the above inequality implies

$$|\{\lambda \in \tilde{L}_t \mid W(H_t^\lambda) \ni w\}| |T^{\lambda, w}(B_{t+1})| y_{K_{t+1}} = O(k_1 d_0^{t_0} d_1^{t_1+1} / |W_t|).$$

Combining this with (35), we have

$$\begin{aligned} \Pr[w \in W(H_t^\lambda)] &= \frac{|\{\lambda \in \tilde{L}_t \mid W(H_t^\lambda) \ni w\}|}{|\tilde{L}_t|} \\ &= \tilde{O} \left( \frac{k_1 d_0^{t_0} d_1^{t_1+1} / (|W_t| |T^{\lambda, w}(B_{t+1})| y_{K_{t+1}})}{k_0 d_0^{t_0+1} d_1^{t_1} / (|T^\lambda(B_{t+1})| y_{K_{t+1}})} \right) \\ &= \tilde{O} \left( \frac{|T^\lambda(B_{t+1})| / |T^{\lambda, w}(B_{t+1})|}{|W_t|} \right), \quad (\text{since } k_0 d_0 \approx k_1 d_1) \end{aligned}$$

which by (34) is what we needed to prove.  $\square$

**Lemma 7.5.** *For uniformly chosen  $\lambda \in \tilde{L}_t$ , for every  $e \in E_t$  we have  $\Pr_\lambda[e \in E(H_t^\lambda)] = \tilde{O}(|E(H_t)|/|E_t|)$ .*

*Proof.* Take any edge  $e \in S_t$ . By repeated applications of Constraint (14), we have

$$\sum_{\lambda \in \tilde{L}_t: E(H_t^\lambda) \ni e} \sum_{K \in T_e^\lambda(B_t)} y_K \leq d_0^{t_0} d_1^{t_1} y_e.$$

As before, for any  $\lambda \in \tilde{L}_t$  such that  $E(H_t^\lambda) \ni e$ , the number of caterpillars this tuple of leaves contributes,  $|T_e^\lambda(B_{t+1})|$ , is roughly uniform (up to a polylogarithmic factor). Thus, we have

$$|\{\lambda \in \tilde{L}_t \mid e \in E(H_t^\lambda)\}| |T_e^\lambda(B_{t+1})| y_{K_{t+1}} \approx \sum_{\lambda \in \tilde{L}_t: E(H_t^\lambda) \ni e} \sum_{K \in T_e^\lambda(B_{t+1})} y_K \leq d_0^{t_0} d_1^{t_1} y_e.$$

By Constraints (13) and (14) and bucketing, we have  $y_e = O(k_0 d_0/|E_t|)$ , so the above inequality implies

$$|\{\lambda \in \tilde{L}_t \mid E(H_t^\lambda) \ni e\}| |T_e^\lambda(B_{t+1})| y_{K_{t+1}} = O(k_0 d_0^{t_0+1} d_1^{t_1} / |E_t|).$$

Combining this with (35), we have

$$\begin{aligned}
\Pr[e \in E(H_t^\lambda)] &= \frac{|\{\lambda \in \tilde{L}_t \mid E(H_t^\lambda) \ni e\}|}{|\tilde{L}_t|} \\
&= \tilde{O}\left(\frac{k_0 d_0^{t_0+1} d_1^{t_1} / (|E_t| |T_e^\lambda(B_{t+1})| y_{K_{t+1}})}{k_0 d_0^{t_0+1} d_1^{t_1} / (|T^\lambda(B_{t+1})| y_{K_{t+1}})}\right) \\
&= \tilde{O}\left(\frac{|T^\lambda(B_{t+1})| / |T_e^\lambda(B_{t+1})|}{|E_t|}\right),
\end{aligned}$$

which by (32) is what we needed to prove.  $\square$

## 7.2 LP rounding: approximation guarantee

The analysis of the approximation guarantee is similar to the combinatorial analysis of Dks in [BCC<sup>+</sup>10], adapted to smES and slightly simplified by the regularity of degrees and LP values which we get from bucketing.

Let us extend the previous notation by letting  $S_t^\lambda$  be the set of vertices in  $S_t$  which serve as rightmost endpoints of caterpillars in  $T^\lambda(B_t)$ . We will show that a certain invariant on the cardinalities and total LP values of the sets  $S_t^\lambda$  is maintained at every step  $t$ , as long as the required conditions for faithful rounding (i.e., the conditions for Step 1 or for Step 3 of the algorithm) do not hold. The proof works by showing that if the invariant holds after the last iteration, then we have arrived at a contradiction. We will consider separately backbone steps and hair steps. Let us begin with backbone steps.

**Lemma 7.6.** *Suppose at iteration  $t$  of Algorithm **Faithful-SmES** the vertices in  $S_t$  are on side  $b$  (where  $b \in \{0, 1\}$ ). Let  $\lambda \in \tilde{L}_t$ . Then if for some  $\beta \leq 1 - \alpha$  the following two conditions hold*

1.  $|S_t^\lambda| y_{K_t} / y_\lambda = \tilde{\Omega}\left(\frac{d_{1-b}}{f} \cdot f^{\beta/\alpha}\right)$  (where  $K_t$  is any caterpillar in  $B_t$ ), and
2.  $|S_t^\lambda| = \tilde{O}(n^\beta)$ ,

then either the conditions for Step 1 hold, or the conditions for Step 3 hold, or the following conditions hold:

1.  $|S_{t+1}^\lambda| y_{K_{t+1}} / y_\lambda \geq 2 \cdot \frac{d_b}{f} \cdot f^{(\beta+\alpha)/\alpha}$  (where  $K_{t+1}$  is any caterpillar in  $B_{t+1}$ ), and
2.  $|S_{t+1}^\lambda| = \tilde{O}(n^{\beta+\alpha})$ .

We note that the last iteration (iteration  $s$ ) of the algorithm is always a backbone step (by our construction). We will show, in the end, that at the last step we have  $\beta = 1 - \alpha$ , and thus (assuming we do not get a faithful factor  $\tilde{O}(f)$  rounding at any point), the above lemma guarantees that  $|S_{s+1}^\lambda| y_{K_{s+1}} / y_\lambda \geq 2d_b \cdot f^{(1-\alpha)/\alpha} = 2d_b (k_{1-b} f / d_b)^{1-\alpha} = 2d_b (k_{1-b} f / d_b) / f = 2k_{1-b}$ . Taking  $K_{s+1}$  to be the caterpillar in  $B_{t+1}^\lambda$  with the smallest  $y_{K_{s+1}}$  value, we get that

$$\sum_{u \in S_{t+1}^\lambda} y_{\lambda \cup \{u\}} / y_\lambda \geq |S_{s+1}^\lambda| y_{K_{s+1}} / y_\lambda \geq 2k_{1-b},$$

contradicting Constraint (13). Let us now prove the above lemma.



*Proof of Lemma 7.6.* By our construction, we know that for a  $\tilde{\Omega}(1)$ -fraction of vertices  $u \in S_t^\lambda$ , there is some caterpillar  $K_u \in T_u^\lambda(B_t)$  such that

$$\sum_{w \in \Gamma(u): K_u \cup \{(u,w)\} \in B_{t+1}} y_{K_u \cup \{(u,w)\}} = \tilde{\Omega}(d_b y_{K_u}).$$

In particular, this gives the following bound on the total “relative LP value” of edges:

$$\sum_{(u,w) \in E(H_t^\lambda)} y_{K_u \cup \{(u,w)\}} / y_\lambda = \tilde{\Omega}\left(\frac{d_0 d_1}{f} \cdot f^{\beta/\alpha}\right). \quad (36)$$

By Lemma B.2 and Proposition 6.1, we may assume that all the degrees in  $G_t = (S_t, W_t, E_t)$ , and thus also in  $H_t^\lambda$ , are at most  $D = \tilde{O}(nd_0/(k_1 f^2)) = \tilde{O}(n^\alpha)$ , and therefore  $|S_{t+1}| \leq |S_t| \cdot D = \tilde{O}(Cn^{\beta+\alpha})$ . Thus, in this case, we only need to prove that (there exists a faithful factor  $f$  rounding or)  $|S_{t+1}^\lambda| y_{K_{t+1}} / y_\lambda \geq 2 \cdot \frac{d_b}{f} \cdot f^{(\beta+\alpha)/\alpha}$ . Suppose the latter condition is not satisfied. Then by (36), the average degree in  $H_t^\lambda$  of vertices in  $W(H_t^\lambda)$  is at least

$$\tilde{\Omega}\left(\frac{d_0 d_1}{f} \cdot f^{\beta/\alpha} / \left(\frac{d_b}{f} \cdot f^{(\beta+\alpha)/\alpha}\right)\right) = \tilde{\Omega}(d_{1-b}/f).$$

Since, as is easy to see, the average degree of vertices in  $U(H_t^\lambda)$  is  $\tilde{\Omega}(d_b) \geq \tilde{\Omega}(d_b/f)$ , by Corollary 5.6 it suffices to show that  $|U(H_t^\lambda)| = \tilde{O}(k_b f)$  (and then by the corollary, there is a faithful factor  $f$  rounding). However, this follows directly:

$$\begin{aligned} |U(H_t^\lambda)| &\leq |S_t^\lambda| \leq n^\beta \leq \text{polylog}(n) \cdot n^\beta \leq \text{polylog}(n) \cdot n^{1-\alpha} \\ &\leq \text{polylog}(n) \cdot k_b f^2 / d_{1-b} \quad \text{by Proposition (6.1)} \\ &\leq \text{polylog}(n) \cdot k_b f. \end{aligned}$$

□

We also have the following lemma for hair steps:

**Lemma 7.7.** *Suppose at iteration  $t$  of Algorithm Faithful-SmES the vertices in  $S_t$  are on side  $b$  (where  $b \in \{0, 1\}$ ). Let  $\lambda \in \tilde{L}_t$ . Then if for some  $\beta \geq 1 - \alpha$  the following two conditions hold*

1.  $|S_t^\lambda| y_{K_t} / y_\lambda = \tilde{\Omega}\left(\frac{d_{1-b}}{f} \cdot f^{\beta/\alpha}\right)$  (where  $K_t$  is any caterpillar in  $B_t$ ), and
2.  $|S_t^\lambda| = \tilde{O}(n^\beta)$ ,

*then either the conditions for Step 1 hold, or the conditions for Step 3 hold, or the following conditions also hold for all leaf tuples  $\lambda' \in L_{t+1}$  which extend  $\lambda$ :*

1.  $|S_{t+1}^{\lambda'}| y_{K_{t+1}} / y_{\lambda'} = \tilde{\Omega}\left(\frac{d_{1-b}}{f} \cdot f^{(\beta-(1-\alpha))/\alpha}\right)$  (where  $K_{t+1}$  is any caterpillar in  $B_{t+1}$ ), and
2.  $|S_{t+1}^\lambda| \leq \frac{1}{2} n^{\beta-(1-\alpha)}$ .

*Proof.* As in the proof of Lemma 7.6, the bound on the total “relative LP value” of edges given by (36) holds. By Constraint (13), we also have  $\sum_{w \in W(H_t^\lambda)} y_{\lambda \cup \{w\}} / y_\lambda \leq k_{1-b}$ . Together, these

inequalities give us the following lower bound on the average “LP-degree” of vertices  $w \in W(H_t)$  (which by our construction is uniform up to a constant factor):

$$\begin{aligned}
\sum_{u:(u,w) \in E(H_t^\lambda)} y_{K_u \cup (u,w)} / y_{\lambda \cup \{w\}} &\geq \frac{1}{\text{polylog}(n)} \cdot \frac{d_0 d_1}{f^{k_{1-b}}} \cdot f^{\beta/\alpha} \\
&= \frac{1}{\text{polylog}(n)} \cdot \frac{d_{1-b}}{f} \cdot f^{(\beta-(1-\alpha))/\alpha} \cdot \left( \frac{d_b}{k_{1-b} f} \cdot f^{1/\alpha} \right) \\
&= \frac{1}{\text{polylog}(n)} \cdot \frac{d_{1-b}}{f} \cdot f^{(\beta-(1-\alpha))/\alpha}.
\end{aligned} \tag{37}$$

Thus, condition 1 always holds. Thus, we need to show that, if there is no faithful factor  $f$  rounding, the average degree (in fact maximum degree, but these are only a constant factor apart) in  $H_t^\lambda$  of vertices  $w \in W(H_t^\lambda)$  is at most  $\frac{1}{2}n^{\beta-(1-\alpha)}$ . Suppose not. That is, suppose the average degree of such vertices is  $\Omega(n^{\beta-(1-\alpha)})$ , and let us show that we can get a faithful rounding of factor  $f$ .

Note that by (37), the average degree is at least  $\tilde{\Omega}(\frac{d_{1-b}}{f} \cdot f^{(\beta-(1-\alpha))/\alpha}) \geq \tilde{\Omega}(\frac{d_{1-b}}{f})$ . Since, as before, it is easy to see that the average degree of vertices  $u \in U(H_t^\lambda)$  is at least  $\tilde{\Omega}(d_b) \geq \tilde{\Omega}(d_b/f)$ , by Corollary 5.5 we only need to show that  $d'_{1-b}/|U(H_t^\lambda)| \geq d_0/(k_1 f^2)$ , where  $d'_{1-b}$  denotes the average degree of vertices  $w \in W(H_t^\lambda)$ . Indeed, by our assumption about this average degree, we have

$$\frac{d'_{1-b}}{|U(H_t^\lambda)|} \geq \Omega\left(\frac{n^{\beta-(1-\alpha)}}{|U(H_t^\lambda)|}\right) \geq \Omega\left(\frac{n^{\beta-(1-\alpha)}}{|S_t^\lambda|}\right) = \tilde{\Omega}\left(n^{-(1-\alpha)}\right) \geq \frac{d_0}{k_1 f^2},$$

where the last inequality follows from Proposition (6.1).  $\square$

Combining these two lemmas, the main theorem easily follows:

**Theorem 7.8.** (Assuming  $f \leq d_0$ .) *Algorithm Faithful-SmES gives either a factor  $f$  faithful rounding for SmES, or it gives a skewed proportional rounding satisfying the conditions of Lemma 5.4.*

*Proof.* Denote by  $b_t$  the side which contains the vertices in  $S_t$ . The theorem follows from the following simple claim, which can be proved directly by induction using the above lemmas (we use the notation  $\{x\} = x + 1 - \lceil x \rceil$ ):

**Claim 7.9.** *Suppose the conditions for Step 1 never hold. Then for all  $t = 2, \dots, s + 1$ , either the conditions for Step 3 hold at at least one of the iterations through step  $t - 1$ , or we have*

1.  $|S_t^\lambda| y_{K_t} / y_\lambda = \tilde{\Omega}(\frac{d_{1-b_t}}{f} \cdot f^{\{(t-1)\alpha\}/\alpha})$  (for all  $\lambda \in L_t$ , and caterpillar  $K_t \in T^\lambda(B_t)$ ), and
2.  $|S_t^\lambda| = \tilde{O}(n^{\{(t-1)\alpha\}})$ .

Moreover, if  $\{(t-1)\alpha\} \geq \alpha$  (i.e. step  $t-1$  was a backbone step), then

$$\sum_{u \in S_t^\lambda} y_{\lambda \cup \{u\}} / y_\lambda \geq 2 \cdot \frac{d_{1-b_t}}{f} \cdot f^{\{(t-1)\alpha\}/\alpha}.$$

As pointed out earlier, this last inequality yields a contradiction for  $t = s + 1$ . Therefore one of the steps gives a factor- $f$  faithful rounding.  $\square$

## 8 Discussion and Future Directions

Some features of our techniques might be applicable to other problems. Most obviously, this is perhaps the first time that LP hierarchies are applied to “local” parts of an LP, rather than to the entire LP. Can this approach be useful for other problems? Currently, it is not clear to us how this approach fares against one “global” application of an LP hierarchy to some basic relaxation: a global hierarchy could take advantage of non-locality in the constraints and solution, but on the other hand would not allow us to locally “guess” degrees (see e.g. footnote 4).

Persistent gaps in the approximability of other network design problems naturally call for a judicious use of LP hierarchies in order to obtain better approximation algorithms. For example, the BASIC  $k$ -SPANNER problem, in which the goal is to construct a  $k$ -spanner with as few edges as possible, is only known to admit approximation ratio  $O(n^{\lceil 2/(k+1) \rceil})$  [ADD<sup>+</sup>93], while the best hardness of approximation is  $2^{(\log^{1-\varepsilon} n)/k}$  for arbitrarily small constant  $\varepsilon > 0$  [DKR12]. An integrality gap that almost matches the upper bound (namely a gap of  $n^{\Omega(1/k)}$ ) was recently shown by Dinitz and Krauthgamer [DK11a], but stronger relaxations obtained via hierarchies can possibly have smaller integrality gaps. In particular, it is not at all clear what the best achievable approximation ratio is for the regime when  $k$  is constant; perhaps hierarchies will finally allow upper bounds that beat [ADD<sup>+</sup>93]. Similarly, for directed  $k$ -spanner the known upper bound is  $\tilde{O}(\sqrt{n})$  [BBM<sup>+</sup>11], and there is an  $\tilde{\Omega}(n^{1/3})$  integrality gap [DK11a], but it only applies to a simple LP relaxation. Yet other relevant problems are DIRECTED STEINER TREE and DIRECTED STEINER FOREST, see [FKN09, BBM<sup>+</sup>11] and references therein. Perhaps hierarchies could help for any of these problems?

Finally, the connection we show between LD2S and *smES* suggests an intriguing possibility for conditional lower bounds. The current hardness for LD2S is only  $\Omega(\log n)$ , while *smES* is basically as hard as *DkS*, which is commonly thought to be difficult to approximate well (say within a polylogarithmic factor, although current hardness results rely on various complexity assumptions and give only a relatively small constant [Fei02, Kho06]). A reduction in the other direction, i.e. from *smES* to LD2S, could give partial evidence that LD2S cannot be approximated well, and could possibly even match the upper bound that we prove here. The same arguments about a formal connection to *DkS* obviously apply also to other network design problems, such as BASIC  $k$ -SPANNER.

## References

- [ABL02] S. Arora, B. Bollobás, and L. Lovász. Proving integrality gaps without knowing the linear program. In *43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 313–322, 2002.
- [ABP92] B. Awerbuch, A. Baratz, and D. Peleg. Efficient broadcast and lightweight spanners. Technical Report CS92-22, Weizmann Institute of Science, 1992.
- [ADD<sup>+</sup>93] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9(1):81–100, 1993.
- [AGGN10] B. Anthony, V. Goyal, A. Gupta, and V. Nagarajan. A plant location guide for the unsure: Approximation algorithms for min-max location problems. *Math. Oper. Res.*, 35(1):79–101, February 2010.
- [AP95] B. Awerbuch and D. Peleg. Online tracking of mobile users. *J. ACM*, 42(5):1021–1058, 1995.

- [BBM<sup>+</sup>11] P. Berman, A. Bhattacharyya, K. Makarychev, S. Raskhodnikova, and G. Yaroslavtsev. Improved approximation for the directed spanner problem. In *38th International Colloquium on Automata, Languages and Programming*, volume 6755 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2011.
- [BCC<sup>+</sup>10] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an  $O(n^{1/4})$  approximation for densest  $k$ -subgraph. In *42nd ACM Symposium on Theory of Computing*, pages 201–210, 2010.
- [BCG09] M. Bateni, M. Charikar, and V. Guruswami. Maxmin allocation via degree lower-bounded arborescences. In *41st annual ACM symposium on Theory of computing*, pages 543–552, New York, NY, USA, 2009. ACM.
- [BGJ<sup>+</sup>09] A. Bhattacharyya, E. Grigorescu, K. Jung, S. Raskhodnikova, and D. P. Woodruff. Transitive-closure spanners. In *20th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 932–941, 2009.
- [BRR10] P. Berman, S. Raskhodnikova, and G. Ruan. Finding Sparser Directed Spanners. In *FSTTCS 2010*, volume 8 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 424–435, 2010.
- [BRS11] B. Barak, P. Raghavendra, and D. Steurer. Rounding semidefinite programming hierarchies via global correlation. In *52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS’11)*. IEEE, 2011.
- [Chl07] E. Chlamtac. Approximation algorithms using hierarchies of semidefinite programming relaxations. In *48th Annual IEEE Symposium on Foundations of Computer Science*, pages 691–701. IEEE Computer Society, 2007.
- [CKR10] E. Chlamtac, R. Krauthgamer, and P. Raghavendra. Approximating sparsest cut in graphs of bounded treewidth. In *13th International Workshop on Approximation, Randomization, and Combinatorial Optimization*, volume 6302 of *Lecture Notes in Computer Science*, pages 124–137. Springer, 2010.
- [CS08] E. Chlamtac and G. Singh. Improved approximation guarantees through higher levels of SDP hierarchies. In *11th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 49–62. Springer-Verlag, 2008.
- [CT12] E. Chlamtac and M. Tulsiani. Convex relaxations and integrality gaps. In M. F. Anjos and J. B. Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, volume 166 of *International Series in Operations Research & Management Science*, pages 139–169. Springer, 2012.
- [Din07] M. Dinitz. Compact routing with slack. In *26th annual ACM Symposium on Principles of Distributed Computing*, pages 81–88. ACM, 2007.
- [DK11a] M. Dinitz and R. Krauthgamer. Directed spanners via flow-based linear programs. In *43rd Annual ACM Symposium on Theory of Computing*, pages 323–332. ACM, 2011.
- [DK11b] M. Dinitz and R. Krauthgamer. Fault-tolerant spanners: better and simpler. In *30th Annual ACM Symposium on Principles of Distributed Computing*, pages 169–178. ACM, 2011.
- [DKR12] M. Dinitz, G. Kortsarz, and R. Raz. Label cover instances with large girth and the hardness of approximating basic  $k$ -spanner. Available at <http://arxiv.org/abs/1203.0224>, 2012.
- [EEST08] M. Elkin, Y. Emek, D. A. Spielman, and S.-H. Teng. Lower-stretch spanning trees. *SIAM J. Comput.*, 38(2):608–628, 2008.

- [EP01] M. Elkin and D. Peleg. The client-server 2-spanner problem with applications to network design. In *8th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 117–132, 2001.
- [Fei02] U. Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the 34th annual ACM Symposium on Theory of Computing (STOC'02)*, pages 534–543. ACM Press, 2002.
- [FKN09] M. Feldman, G. Kortsarz, and Z. Nutov. Improved approximating algorithms for directed Steiner forest. In *20th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 922–931. SIAM, 2009.
- [FS97] U. Feige and M. Seltser. On the densest  $k$ -subgraph problem. Technical report, Weizmann Institute of Science, Rehovot, Israel, 1997.
- [GHN07] A. Gupta, M. Hajiaghayi, V. Nagarajan, and R. Ravi. Dial a ride from  $k$ -forest. In L. Arge, M. Hoffmann, and E. Welzl, editors, *Algorithms ESA 2007*, volume 4698 of *Lecture Notes in Computer Science*, pages 241–252. Springer Berlin / Heidelberg, 2007.
- [GS11] V. Guruswami and A. K. Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with PSD objectives. In *52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS'11)*. IEEE, 2011.
- [Kho06] S. Khot. Ruling out PTAS for graph min-bisection, dense  $k$ -subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4):1025–1071, 2006.
- [Kor01] G. Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30(3):432–450, 2001.
- [KP93] G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *34th Annual Symposium on Foundations of Computer Science*, pages 692–701, 1993.
- [KP94] G. Kortsarz and D. Peleg. Generating sparse 2-spanners. *J. Algorithms*, 17(2):222–236, 1994.
- [KP98] G. Kortsarz and D. Peleg. Generating low-degree 2-spanners. *SIAM J. Comput.*, 27(5):1438–1456, 1998.
- [Las02] J. B. Lasserre. An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. *SIAM Journal on Optimization*, 12(3):756–769, 2002.
- [LNSS09] L. C. Lau, J. S. Naor, M. R. Salavatipour, and M. Singh. Survivable network design with degree or order constraints. *SIAM J. Comput.*, 39(3):1062–1087, 2009.
- [LS91] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(12):166–190, 1991.
- [Nut10] Z. Nutov. Approximating steiner networks with node-weights. *SIAM Journal on Computing*, 39(7):3001–3022, 2010.
- [PS89] D. Peleg and A. A. Schäffer. Graph spanners. *J. Graph Theory*, 13(1):99–116, 1989.
- [PU89] D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18:740–747, August 1989.
- [SA90] H. D. Sherali and W. P. Adams. A hierarchy of relaxation between the continuous and convex hull representations. *SIAM J. Discret. Math.*, 3(3):411–430, 1990.
- [SL07] M. Singh and L. C. Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC '07: Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, pages 661–670, New York, NY, USA, 2007. ACM.

- [ST04] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *36th Annual ACM Symposium on Theory of Computing*, pages 81–90. ACM, 2004.
- [TZ01] M. Thorup and U. Zwick. Compact routing schemes. In *13th annual ACM Symposium on Parallel Algorithms and Architectures*, pages 1–10. ACM Press, 2001.
- [TZ05] M. Thorup and U. Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005.

## A Handling small degrees

Just as in [BCC<sup>+</sup>10], we may assume that the subgraph degrees (in the optimum, or according to the LP) are greater than or equal to our desired approximation ratio  $f$ . The reason is that there is always a simple  $d_0$ -approximation (recall our convention that  $d_0 \leq d_1$ ). Thus if  $d_0 \leq f$ , we are done.

**Lemma A.1.** *There exists a faithful rounding with factor  $\tilde{O}(d_0)$ .*

*Proof.* Consider the following algorithm. Divide all edges in the graph into buckets by their LP-values, and also by the LP values of their endpoint vertices (so that all three parameters are uniform up to a constant factor within each bucket). Then there is some bucket  $B$  for which  $\sum_{e \in B} y_e = \tilde{\Omega}(m)$ . For  $b = 0, 1$ , let  $U_b$  be the set of vertices on side  $b$  which have at least one edge in  $B$  incident to them. Now pick a subset  $U'_1 \subseteq U_1$  of size  $k_1$  uniformly at random, and let  $B(U'_1)$  be the set of edges in  $B$  incident to vertices in  $U'_1$ . Finally, choose  $B' \subseteq B(U'_1)$  of size

$$|B(U'_1)| \cdot \frac{m}{|B|} \cdot \frac{|U_1|}{k_1} \tag{38}$$

uniformly at random. We return the graph induced by the edges in  $B'$ .

Note that by uniformity of LP values in  $B$ , every edge  $e \in B$  has LP weight  $y_e = \tilde{\Theta}(m/|B|)$ . Moreover, since by Constraint (14), vertices on side  $b$  (for  $b = 0, 1$ ) all contribute LP degree (that is, the LP weight of its incident edges divided by its LP value)  $\tilde{\Theta}(d_b)$ , then the LP weight of every vertex  $u \in U_b$  is  $\tilde{\Theta}(k_b/|U_b|)$ . For brevity, let us omit polylogarithmic factors in the remaining discussion. Note that since for any vertex  $u$  and edge  $e$  incident to  $u$  we have  $y_e \leq y_u$ , this implies

$$\frac{m}{|B|} \leq \frac{k_1}{|U_1|}, \tag{39}$$

which also shows that the quantity in (38) is at most  $|B(U'_1)|$ .

For vertices  $u_1 \in U_1$ , the probability that  $u_1 \in U'_1$  is  $k_1/|U_1| = y_{u_1}$ . This, together with the fact that  $|U'_1| = k_1$  gives faithfulness for vertices in  $U_1$  even for approximation factor 1. Moreover, for any edge  $e \in B$ , the probability that  $e \in B(U'_1)$  is also  $k_1/|U_1|$ . Conditioned on  $e \in B$ , the probability that  $e$  is retained in  $B'$  is  $\frac{m}{|B|} \cdot \frac{|U_1|}{k_1}$ , and thus the probability (a priori) that we will have  $e \in B'$  is  $m/|B| = y_e$ , which gives faithfulness for edges.

Next, we would like to show that at most  $m$  edges are chosen, which will also give the required bound on the number of vertices in  $U_0$  chosen, since  $m = d_0 k_0$ . Note that since the LP degree of every vertex  $u_1 \in U_1$  is at most  $d_1$ , by the LP values of edges and vertices in  $B$ , this implies that the graph degree of every vertex in  $U_b$  (for  $b = 0, 1$ ) is at most

$$D_b = d_b \cdot \frac{k_b/|U_b|}{m/|B|}. \tag{40}$$

Thus, the total number of edges in  $B(U'_1)$  is always at most  $|U'_1| \cdot d_1 k_1 |B| / (|U_1| m)$ . By (38), the number of edges in  $B'$  is then always at most  $|U'_1| \cdot d_1 = k_1 d_1 = m$ .

It remains to analyze the probability that an individual vertex in  $U_0$  is chosen. Since every vertex  $u_0 \in U_0$  is picked iff one of its incident edges is included in  $B'$ , and since the probability of each such event is at most  $y_e$ , by a union bound, and by (40), the probability that  $u_0$  is picked is at most

$$D_0 y_e = d_0 \cdot \frac{k_0 / |U_0|}{m / |B|} \cdot y_e = d_0 \cdot \frac{k_0 / |U_0|}{m / |B|} \cdot \frac{m}{|B|} = d_0 \cdot \frac{k_0}{|U_0|} = d_0 \cdot y_{u_0}.$$

□

## B Bounding the maximum degree

It remains to show the correctness of Step 1 in Algorithm **Faithful-SmES**. Before we do this, let us start with a simple claim.

**Claim B.1.** *In the graph  $G_t$ , the average degree and maximum degree of vertices in  $W_t$  differ by at most a polylogarithmic factor (and the same holds for vertices in  $S_t$ ).*

*Proof.* Let us show this for  $W_t$  (the proof for  $S_t$  is identical). By Lemma 6.3, the LP degree of the average vertex  $w \in W_t$  (that is, the quantity  $\sum_{u:(u,w) \in E_t} y_{\{u,w\}} / y_{\{w\}}$ ) is  $\Omega(d_1)$ , and by Constraint (14) no LP-degree (for vertices in  $W_t$ ) can be more than  $d_1$ . Moreover, by uniformity of LP values, the LP-degrees are proportional to the graph degrees, which proves the claim. □

**Lemma B.2.** *If at any iteration  $t$  Algorithm **Faithful-SmES**, the maximum degree in  $G_t$  is at least  $\tilde{\Omega}\left(\frac{nd_0}{k_1 f^2}\right)$ , then Step 1 gives a faithful factor- $f$  rounding.*

*Proof.* Without loss of generality, assume  $S_t \subseteq V_0$  (the case where  $S_t \subseteq V_1$  is essentially the same). By Lemma 6.3 and uniformity of LP values (which we get from bucketing), we have that for every vertex  $u \in S_t$ ,  $y_{\{u\}} \approx k_0 / |S_t|$ , for every vertex  $w \in W_t$ ,  $y_{\{w\}} \approx k_1 / |W_t|$ , and for every edge  $e \in E_t$ ,  $y_{\{e\}} \approx m / |E_t|$ .

Thus, it is clear that the random sampling of vertices in Step 1 is faithful with respect to vertices. It suffices, then, to show that every edge  $e \in E_t$  is picked with probability  $\tilde{\Theta}(y_{\{e\}})$  (this is sufficient since they have total LP weight  $\tilde{\Omega}(m)$ ). Indeed, since every vertex  $v \in S_t \cup W_t$  is chosen with probability  $\tilde{\Theta}(y_{\{v\}} f)$ , the probability that both endpoints of an edge in  $e = (u, w) \in E_t$  are chosen is  $\tilde{\Theta}(y_{\{u\}} y_{\{w\}} f^2)$ . Hence we only need to show that this probability is at least  $y_{\{e\}}$ . This follows since, by the maximum degree bound, and by Claim B.1, we have that (ignoring polylogarithmic factors):

$$|E_t| \geq |W_t| \cdot \frac{nd_0}{k_1 f^2} \approx \frac{nd_0}{y_{\{w\}} f^2} \approx \frac{nm}{y_{\{w\}} f^2 k_0} \geq \frac{|S_t| m}{y_{\{w\}} f^2 k_0} \approx \frac{m}{y_{\{u\}} y_{\{w\}} f^2} \approx \frac{y_{\{e\}} |E_t|}{y_{\{u\}} y_{\{w\}} f^2}.$$

□