Hardness and Approximation Algorithms for Balanced Districting Problems

Prathamesh Dharangutte* Jie Gao* Shang-En Huang† Fang-Yi Yu‡

Abstract

We introduce and study the problem of balanced districting, where given an undirected graph with vertices carrying two types of weights (different population, resource types, etc) the goal is to maximize the total weights covered in vertex disjoint districts such that each district is a star or (in general) a connected induced subgraph with the two weights to be balanced. This problem is strongly motivated by political redistricting, where contiguity, population balance, and compactness are essential. We provide hardness and approximation algorithms for this problem. In particular, we show NP-hardness for an approximation better than $n^{1/2-\delta}$ for any constant $\delta > 0$ in general graphs even when the districts are star graphs, as well as NP-hardness on complete graphs, tree graphs, planar graphs and other restricted settings. On the other hand, we develop an algorithm for balanced star districting that gives an $O(\sqrt{n})$-approximation on any graph (which is basically tight considering matching hardness of approximation results), an $O(\log n)$ approximation on planar graphs with extensions to minor-free graphs. Our algorithm uses a modified Whack-a-Mole algorithm [Bhattacharya, Kiss, and Saranurak, SODA 2023] to find a sparse solution of a fractional packing linear program (despite exponentially many variables) which requires a new design of a separation oracle specific for our balanced districting problem. To turn the fractional solution to a feasible integer solution, we adopt the randomized rounding algorithm by [Chan and Har-Peled, SoCG 2009]. To get a good approximation ratio of the rounding procedure, a crucial element in the analysis is the *balanced scattering separators* for planar graphs and minor-free graphs — separators that can be partitioned into a small number of k-hop independent sets for some constant k — which may find independent interest in solving other packing style problems. Further, our algorithm is versatile — *the very same algorithm* can be analyzed in different ways on various graph classes, which leads to class-dependent approximation ratios. We also provide a FPTAS algorithm for complete graphs and tree graphs, as well as greedy algorithms and approximation ratios when the district cardinality is bounded, the graph has bounded degree or the weights are binary.

*Department of Computer Science, Rutgers University. Email: {prathamesh.d,jg1555}@rutgers.edu. Dharangutte and Gao are supported by NSF IIS-2229876, DMS-2220271, DMS-2311064, CCF-2208663, CCF-2118953.

†National Taiwan University. Email: math.tmt514@gmail.com. Shang-En Huang conducted part of this research in Boston College and in visiting Osaka University. Shang-En is currently supported by NTU Grant 113L7491.

‡George Mason University, Email: fangyiyu@gmu.edu

1 Introduction

In this paper we study the problem of balanced districting, where we are given an undirected graph where vertices carry two types of weights (population, resource types, etc) and the goal is to find vertex disjoint districts such that each district is a connected induced subgraph with the total weights to be c-balanced – each type of weight is at least $1/c$ of total weight of the district. We aim to maximize the total weights of the vertex disjoint balanced districts.

This problem is an abstraction of many real world scenarios of districting where contiguity/connectivity, population/resource type balance and compactness are desirable properties. For example, in political redistricting, several towns are grouped into a state legislative district or a congressional district. Balancedness requires that each district maintains a sufficient fraction of each (political or demographic) group, which is essential for several reasons. First, voter turnout rates sharply increase if the anticipated election outcome is expected to be a close tie. [16] Thus a balanced district would motivate and raise voter turnout rates. Additionally, balancedness ensures that each political group has the opportunity to elect a candidate of their choice, in compliance with the Voting Rights Act of 1965 [48] and other amendments [29]. This principle also helps to prevent the tipping point in racial segregation, where residents of one demographic group start to leave a district once their population falls below a certain threshold. [65, 61] Connectivity or contiguity, on the other hand, demands that each district be geographically contiguous, – in the language of graph theory that the vertices corresponding to the towns form an induced connected subgraph. This requirement is enforced by most state laws and is a standard practice in general. Many states also have a compactness rule [3], which refers to the principle that the constituents residing within an electoral district should live as near to one another as possible. It often manifests into a preference for regular geometric shapes or high roundness (small ratio of circumference and total area).

The problem of redistricting also appears in many other scenarios such as districting for public schools, sales and services, healthcare, police and emergency services [15], and logistics operations [53]. In addition, the balanced districting problem is of interest in a broader range of applications for resource allocation. For example modern computing infrastructure such as cloud computing provides services to a dynamic set of customers with diverse demands. Customer applications may have a variety of requirements on the combination of different resources (such as CPU cycles, memory, storage, or access of special hardware) that can be summarized by the balanced requirement.

Due to the importance of the problem, redistricting has been studied in a computational sense for schools and elections, that dates back to the 1960s. [49] Since then, an extensive line of work (see [9] for a survey) has formulated the redistricting task as an optimization problem with a certain set of objectives. A lot of existing work considers the geographical map as input and comes up with practical methods and software implementations that generate feasible districting plans. We will survey such work in Section 1.2. However, most previous works focus on optimizing a single desirable property alone (e.g., connectivity [1], or balance [45]), or optimizing average aggregated scores combining multiple objectives of the districts [35]. In contrast, our problem formulation takes these objectives as hard constraints and optimizes the total population that satisfies them. There are several merits of this formulation. First, it offers interpretable, fair, worst-case guarantees for the identified districts. Districting problem is a multi-faceted one. With multiple criteria taken into consideration, it feels ill-fit if only one criterion is singled out as the optimization objective. Furthermore, an average quality guarantee does not provide meaningful utility at the individual district level, and aggregated scores offer limited insight into each objective. Second, a districting solution has a consequential nature and should be taken with a dynamic and time-evolving perspective. Once a new districting plan is in place, residents naturally respond to the algorithm output, resulting in changes in the population distributions. One prominent example is the tipping point theory in racial segregation mentioned above. Optimizing a single balancedness score can still lead to many districts falling below such a tipping point, exacerbating segregation. With this in consideration it is important to keep balancedness as a hard constraint, which hopefully facilitates district stability and integration. With c-balanced property as a requirement, a graph partitioning into vertex disjoint c-balanced districts is not always possible. For example, if the total weight is not c-balanced, some districts have to be unbalanced no matter how the districts are defined. Therefore, we aim to maximize the total weights in balanced districts.

In this paper we focus on the graph theoretic perspective of the redistricting problem. We abstract the input as a graph where vertices represent natural geographical entities/blocks (e.g., townships) and edges of two vertices represent geographical adjacency/contiguity. We focus on two important quality considerations namely connectivity and balanceness, and we maximize coverage, i.e., the total weight (population) covered by balanced districts. In addition, we also consider compactness, which in our setting leads to preference of districts as low-diameter subgraphs. An important case studied in this paper is to consider a balanced *star district*, which consists of a center vertex v as well as a set of neighboring blocks all adjacent to v. We also consider districts of bounded rank k for a constant k – where a district has at most k vertices.

1.1 Our Results and Technical Overview

We report a systematic study of the balanced districting problem on both hardness results and approximation algorithms. Our goal is to dissect the problem along different types of input graph topologies (general graphs, planar graphs, bounded degree graphs, complete graphs, tree graphs, etc), district types (e.g., arbitrarily connected districts, star districts, or bounded rank-k), and weight assumptions (arbitrary weights, binary weights). A brief summary of our results can be found in Table 1.

Complexity and Challenges. There are three elements in the balanced districting problem that make it challenging and interesting, from a technical perspective: 1) connectivity – the induced subgraph of a district is connected 2) packing and coverage maximization – no vertex belongs to two districts and we maximize the total weights of included vertices; 3) balancedness – the two types of weights in a district need to be roughly balanced. These elements are shared with a number of well known hard problems, suggesting that our problem is also computationally challenging. For example, the exact set cover problem asks if there is a perfect coverage and packing in a set cover instance. The packing element is shared with maximum independent set problem – a vertex included will forbid its neighbors to be included. And the balancedness is shared with subset sum problem. Therefore by using the hardness of these problems we can show hardness and hardness of approximation of the balanced districting problem for a variety of graph classes. The hardness of the balanced districting problem is immediately shown by a reduction from exact set cover problem. By a reduction from maximum independent set problem, we can show that the balanced districting problem does not have an approximation of $n^{1/2-\delta}$ for any constant $\delta > 0$ in a general graph of n vertices unless $\mathsf{P} = \mathsf{NP}$ and is APX-hard for bounded degree graphs. From a reduction from the planar 1-in-3SAT problem, the balanced districting problem is NP-hard for a planar graph, and even if each district has at most three vertices – a crucial condition since the problem can be solved exactly by maximum weighted matching if each district is only allowed to have two vertices. If the input graph is a tree or a complete graph – extremely simple topologies for which many NP-hard problems can be solved in polynomial time, the balanced districting problem is still NP-hard due

Table 1: A summary of hardness and approximation results on the balanced districting problem. Note that $\delta, \varepsilon > 0$ are constants. Tight results are highlighted in bold.

	Graph Type	District Type	Result	Note
Section 3	General	arbitrary/star	**NP-hard for $n^{1/2-\delta}$-approx**	Theorem 4
	Max degree Δ	arbitrary/star	APX-hard for $\Delta = O(1)$ NP-hard for $\Delta/2^{O(\sqrt{\Delta})}$-approx UGC-hard for $O(\Delta/\log^2 \Delta)$-approx	Theorem 4
	Planar with $\Delta = 3$	star with rank-3	NP-hard	Theorem 2
	Complete or Tree	arbitrary/star	**NP-hard**	Theorem 3
Section 4	Planar	star	$O(\log n)$-approx	Theorem 5
	H-Minor-Free	star	$O(h^2 \log n)$-approx, $h = \|H\|$	
	Outerplanar	star	$O(1)$-approx	
	General	star	**$\Theta(\sqrt{n})$-approx**	
Section 5	Complete graph	arbitrary/star	**FPTAS $(1+\varepsilon)$-approx**	Theorem 7
	Tree	arbitrary/star	**FPTAS $(1+\varepsilon)$-approx**	Theorem 8
Section 6	General	rank-2	polynomially solvable	Theorem 9
	General	rank-k, $k > 2$	k-approx	Theorem 10
	Bounded degree Δ	star	$(\Delta + \frac{1}{\Delta})$-approx	Theorem 11
	General graph with binary weights	star	c-approx	Theorem 12

to the balancedness requirement by a reduction from subset sum. All of the hardness proofs hold even if we limit the output districts to be only of a star topology. This set of hardness results can be seen from the Top section of Table 1.

Greedy Methods on Special Cases. On the positive side, it is natural to ask if existing techniques for solving or approximating these related problems can be borrowed for the balanced districting problem. The answer turns out to be often "not really", even if we only look for balanced star districts. The additional requirements in our problem often break some crucial steps. For example, the problem of maximum independent set has an easy $\Omega(n)$ lower bound if the input graph is sparse (or planar) — thus a simple random greedy algorithm with conflict checking gives an easy constant approximation algorithm. But such lower bound no longer holds true for our problem if the weights are not balanced. Even for star districts, when the maximum degree is not bounded by a constant, the number of potential balanced districts can be exponential in n, the size of the network.

We show in Section 6 that ideas using a greedy approach and local search method give us approximation algorithms, but only for very special cases. Namely, if the districts have rank-k, we can try the greedy maximum hypergraph matching to have a k-approximation to the optimal solution. When all weights are binary (1 or 0), we can use a greedy algorithm with local search to get a c approximation to the optimal c-balanced star districting solution. Similarly, if the graph has maximum degree Δ, we can get a $(\Delta + \frac{1}{\Delta})$-approximation for c-balanced star districting.

LP Framework and Rounding. To really tackle the problem with arbitrary weights, for districts that are not limited by rank and graphs beyond constant bounded degree, we first examine what we can do with complete graphs or tree graphs – here the topology is made some of the simplest possible, and we would like to address the challenge from packing and balancedness. In this setting we can obtain FPTAS for both complete graphs and tree graphs (Section 5)– although the algorithm is much more involved due to the additional requirements of packing and connectivity (for tree graphs, as connectivity is trivial for complete graphs). Our FPTAS uses a dynamic programming technique that maintains one district's possible weights and introduces a new prioritized trimming method to approximate weights while ensuring that the resulting district satisfies the c-balanced constraint and approximates the optimal weight. The FPTAS for the complete graph is later used as a subroutine for solving the relaxed LP formulation for other graph settings.

Beyond complete graphs and tree graphs, we develop a general framework (Section 4) that produce approximation algorithms for star districts on different types of graphs. All these algorithms start from a relaxed linear program where we formulate a variable x_S for each potential balanced star district S, which can take non-integer values and for all districts that share the same vertex, the sum of their variables is at most 1. Despite potentially exponentially many variables (and constraints in the dual program) that preclude standard solutions, we adapt the whack-a-mole framework [11], which can be seen as a lazy multiplicative weight update algorithm, on dual variables, and we design a 'separation oracle' that selects a violating constraint in the dual program in time polynomial in n and $1/\varepsilon$ that can significantly improve the solution. Consequentially, the linear program can be solved in time polynomial in n and $1/\varepsilon$ up to any precision $1-\varepsilon$ and the number of non-zero primal variables (i.e., the candidate balanced star districts) is also polynomial. Intriguingly, our separation oracle is based on our FPTAS on compute graphs for balanced districting.

Now we will round the fractional solution to an integer solution and in the process we may lose an approximation factor. We use a simple randomized rounding method where we sort the districts with non-zero values in decreasing order of total weight, and flip a coin with probability proportional to x_S to include a potential district S, if S does not overlap with any districts already included. In order to bound the loss of quality in the rounding process, we need to upper bound the correlation of the variables, namely, sum of $x_A \cdot x_B$ for all pairs of overlapping districts A, B. These are the (fractional) districts that have to be dropped due to conflict. To establish the approximation factor, we wish to bound the total sum of correlation by a factor multiplied with the total sum over all possible districts $\sum_S x_S$ – exactly the optimal LP solution. We show that this ratio is $O(\sqrt{n})$, which immediately gives an $O(\sqrt{n})$-approximate solution for star districts on a general graph. Notice that this is tight due to the hardness of approximation results.

Due to the strong motivation from political redistricting and resource allocation considering geographical proximity/constraints, the planar graph is of particular interest to us. One of the main technical contributions is a polylogarithmic-approximation algorithm for balanced star districting on planar graphs and related algorithms for minor-free graphs and outer planar graphs. For a planar graph we now adopt a balanced planar separator and use a divide-and-conquer analysis. Namely, we only need to analyze the overlapping districts with at least one of them including vertices in the separator. Now a crucial observation is, if we can partition the planar separator into k 5-hop independent sets, then we can decompose the total sum of correlation by the independent sets – fix an 5-hop independent set X, two star districts that touch different vertices in X are disjoint and star districts that share the same vertex in X have their total district value bounded by 1 due to the primal constraint. This allows us to upper bound the correlation term for the star districts touching the separator by a factor of k of the sum of x_S with districts S on the separator. Recursively, this gives an $O(\log n)$ factor loss in the final approximation.

We remark that the above analysis asks for a new property of a balanced separator – one that can be decomposed into a small number of 5-hop independent sets (called a "scattering" separator) – and we do not care about the size of the separator. This is possible for a planar graph if we use the fundamental cycle separator, which is composed of two shortest paths, and thus at most

10 5-hop independent sets. For a H-minor-free graph with H as a graph of h vertices, we show the existence of a similar separator, which can be decomposed into $O(h^2)$ 5-hop independent sets. Thus the final approximation ratio for H-minor-free graphs is $O(h^2 \log n)$. For outer planar graphs, we can skip the recursive step and work with graph partitions with 5-hop independent sets and get $O(1)$-approximation. We believe that this technique of using balanced scattering separators is interesting in its own and may find additional applications in other problems with some packing (non-overlapping) requirement on the solution.

On general graphs, the formulated linear program could have an integrality gap as large as $\Omega(\sqrt{n})$. Since our rounding algorithm turns an fractional solution into an integral one, this barrier unavoidably blends into our analysis to the proposed rounding algorithm, producing an provable $O(\sqrt{n})$ bound. However, by thinking about this argument contrapositively, an upper bound to our rounding algorithm leads to the integrality gap of the formulated LP, which could be an interesting takeaway. On the other hand, we also show that there are planar graphs (specifically grid graphs) such that our rounding algorithm produces a > 1 constant approximation ratio. This observation suggests that we cannot hope for a PTAS using this approach, even on planar graphs.

1.2 Related work

To the best of our knowledge, this paper is the first to study the balanced districting problem. Below, we briefly survey related problems and explore their potential connections to ours.

Districting. Our problem is connected to computational (re)districting for schools and elections, which dates back to the 1960s. [49] Since then, an extensive line of work (see [9] for a survey) has formulated the redistricting task as an optimization problem with a certain objective and constraints, e.g,. balancedness, contingency, or compactness. Our redistricting problem focuses on optimizing the population in balanced and contiguous districts. One concept related to our notion of balance is competitiveness. Recent work introduces vote-band metrics [34], which require a certain fraction of votes to fall within a specified range (e.g., 45-55%) for competitive elections. Subsequently, [25] also adopt similar notions called δ-Vote-Band Competitive which is equivalent to our c-balancedness by setting $c = 2/(1-2\delta)$. While related, our work diverges technically, offering both hardness and algorithmic results for several common graphs. [34] empirically evaluates ensemble methods for district distributions. [25] explored the hardness and heuristic algorithms for maximizing the number of districts meeting the target competitiveness constraints, with additional requirements that all districts have roughly the same population limited compactness consideration.

One approach treats contiguity as a transportation cost and designs linear programming models to minimize this total cost [1, 40, 28]. Interestingly, the fair clustering problem can also be viewed as optimizing contingency [14, 22, 51, 21]. Other research focuses on optimizing compactness scores [7, 52, 55, 50] or using Voronoi or power diagrams with some variant of k-means [67, 30, 31, 41]. Finally, another line of work optimizes balance scores [45]. These approaches differ from ours in that they treat specific aspects of districting (contiguity, compactness, balance, etc.) as objectives, rather than maximizing the population that meets these criteria.

Besides the optimization approach, another popular approach uses sampling to generate a distribution over districts and create a collection of district plans for selection. [4, 27] One widely used method is ReCom [35], an MCMC algorithm. However, these approaches may suffer from slow mixing times and lack formal guarantees. [60] Finally, several papers take a fair division approach [56, 62, 33]. The problem is quite different, however, as fairness is defined concerning parties (types) and the number of seats they would win (i.e., the number of districts where they would have a majority) compared to other districts.

Algorithms. Beyond districting problems, as outlined in the technical overview, our problem connects to several classical algorithm problems. If we only want to maximize the population of a single connected and balanced district, the problem becomes a balanced connected subgraph problem [12, 13, 58]. However, the previous work in this area typically considers unit weights for either type, which does not adequately represent the districting problem that operates in an aggregated block-level setting. Our problem can be seen as packing subgraphs on graph [32], e.g., edges (maximum matching), triangles [54], circles [37]. Finally, we note a line of work on a balanced, connected graph partition [20, 24], and balanced bin-packing problem [39], which, however, aim to generate a partition where each component has similar weights.

1.3 Open Problems

As the first work to formally study the balanced districting problem in this formulation, our work leaves a number of interesting open problems for future work. Obviously it is good to close the gap of approximation and hardness for different families of graphs. Our results are tight for general connected districts on complete graphs and tree graphs, as well as star districts on general graphs, but leave gaps for other settings. We conjecture that there exists an algorithm with a constant approximation factor for c-balanced star districting on planar graphs. It would also be interesting to develop algorithms to go beyond star districts, i.e., k-hop graphs for a constant k or the more general setting of connected districts. We remark that the scattering separator can be modified to handle k-hop graphs but we need an efficient separation oracle. We consider two types of weights/populations and generalizing the problem and solutions to three or more colors would be interesting and is currently widely open. We remark that the PTAS algorithm for complete graphs is specific for two weights/colors. Finally, an interesting future direction would be to develop algorithms that also demand approximate population equality among districts.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph where we call the vertices *blocks*. A *district* $T \subseteq V$ is a subset of blocks where the induced subgraph $G[T]$ is connected. If there exists a block $x \in T$ that is a neighbor of every other block in $T \setminus \{x\}$, then we say T is a *star district* and x is a *center* of T. The *rank* of a district T is the number of blocks in T. A *(partial) districting* $\mathcal{T}$ is a collection of disjoint districts. That is, $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ where $T_i \cap T_j = \emptyset$ whenever $i \neq j$. Notice that a districting is not necessarily a partitioning of the graph, i.e., not all blocks are included in the districts.

c-Balanced Districting Problems. There are two communities or commodities of interest. Let the functions $p_1, p_2 : V \to \mathbb{Z}_{\geq 0}$ represent the *population of each community* or *weight of each commodity* on each vertex. The *weight* of a block $w(x)$ is defined to be $p_1(x) + p_2(x)$. Let $T \subseteq V$ be a district. By a natural extension we define $p_i(T) := \sum_{x \in T} p_i(x)$ for $i \in \{1, 2\}$ and $w(T) := p_1(T) + p_2(T)$ accordingly as the weight of the district T. Finally, given a districting $\mathcal{T}$, we define $w(\mathcal{T}) := \sum_{T \in \mathcal{T}} w(T)$ to be the total weight.

Given a constant $c \geq 2$, we say that a district T is *c-balanced* if

$$\min\{p_1(T), p_2(T)\} \geq \frac{w(T)}{c} \ . \tag{1}$$

$\mathcal{T}$ is a *c-balanced districting* if all districts $T \in \mathcal{T}$ are c-balanced. Notice that if the total weights in the graph are not c-balanced, we cannot hope to include all blocks in c-balanced districts. Given

$c \geq 2$, a graph G, and functions p_1 and p_2, the problem c-BALANCED-DISTRICTING is subjected to find any c-balanced districting $\mathcal{T}$ that *maximizes* $w(\mathcal{T})$. That is, we wish to maximize population covered in the c-balanced districts.

We will investigate several variants to the problem. A lot of our results concern restricting the output districting to be *star shaped*, respecting the need for compactness of the districts. We also consider districts of bounded rank k if every district has at most k vertices with k assumed to be a constant. In general we consider the weights of the vertices to be arbitrary integer values. A special case is when all weights are uniform (binary) – each vertex has only one non-zero weight type, either $p_1(x) = 0$, $p_2(x) = 1$ or $p_1(x) = 1$, $p_2(x) = 0$.

Let X be any variant to the c-balanced districting problem. A districting $\mathcal{T}$ is said to be *feasible* on X if $\mathcal{T}$ satisfies all districting type constraints, but not necessarily to have its weight maximized. Any districting with the maximum possible total weight is said to be *optimal*. We say that a feasible districting $\mathcal{T}$ is an *f-approximated* solution if $f \cdot w(\mathcal{T}) \geq w(\mathcal{T}_{\mathsf{OPT}})$, where $\mathcal{T}_{\mathsf{OPT}}$ is any optimal districting.

Graph Types. A graph $G = (V, E)$ is said to be *planar* if there exists an embedding of all vertices to the Euclidean plane such that all edges can be drawn without intersections other than the endpoints. A *face* of an planar embedding is a connected region separated by the embedded edges. G is said to be *outerplanar* if there exists an embedding of G such that there is a face containing all vertices. Often this face is assumed to be the outer face. A graph H is said to be a *minor* of G if H is isomorphic to the graph obtained by a sequence of vertex deletions, edge contractions, and edge deletions from G. We say that G is *H-minor-free* if G does not have H as its minor.

3 Hardness Results

We first present hardness results for a variety of c-balanced districting problems with increasing restrictions on the parameters. The proofs are deferred to Appendix A.

Theorem 1. *The c-balanced districting problem is* NP*-hard, for both the case when the the districts are connected subgraphs and when the districts are required to be stars.*

Theorem 1 uses a reduction from the EXACTSETCOVER problem. The EXACTSETCOVER problem remains NP-hard even when each set has *exactly* three elements and no element appears in more than three sets [44], or when each element appears in exactly three sets [46]. Therefore, if we limit that each district has at most *four* blocks, the problem remains NP-hard. In the following we show that the problem remains hard if each district has at most *three* blocks and the graph is planar. Notice that if each district has at most two blocks, the problem can be solved by maximum matching in polynomial time.

Theorem 2. *The c-balanced districting problem is* NP*-hard, when G is a planar graph with maximum degree 3, each district has rank-3 (i.e., with at most three blocks), and the districts must be stars.*

The proof of the above claim uses reduction from planar 1-in-3SAT. We show next that the problem on a complete graph or a tree remains hard. This reduction uses the problem of subset sum.

Theorem 3. *The c-balanced districting problem is* NP*-hard for any* $c \geq 2$*, when* G *is a complete graph or a tree. This holds for both the case when the the districts are connected subgraphs and when the districts are required to be stars.*

Last, we show hardness of approximation by a reduction from the maximum independent set problem.

Theorem 4. *The c-balanced districting problem does not have an* $n^{1/2-\delta}$*-approximation (for any constant* $\delta > 0$*) in a general graph unless* $\mathsf{P} = \mathsf{NP}$*. On a graph with maximum degree* Δ*, one cannot approximate the c-balanced districting problem within a factor of* $\Delta/2^{O(\sqrt{\Delta})}$ *assuming* $\mathsf{P} \neq \mathsf{NP}$*, and* $O(\Delta/\log^2 \Delta)$ *assuming the Unique Games Conjecture (UGC). Even if* Δ *is a constant, the problem is* APX*-hard. These statements hold when the districts must be stars and when the centers of the stars are limited to a subset of vertices.*

4 An Algorithm for c-Balanced Star Districting

In this section, we give an approximation algorithm to the c-balanced star districting problem. The algorithm is based on a multiplicative weights update approach of solving packing-covering linear programs [63, 5, 11] and then apply a randomized rounding procedure [17]. Interestingly, the same algorithm achieves different approximation guarantees on different classes of the graphs, summarized in the following theorem.

Theorem 5. *Let* G *be a graph with function of weights* p_1 *and* p_2*. There exists a polynomial time algorithm that computes a c-balanced star districting* $\mathcal{T}$*, with the following guarantee:*

(1) For any general graph G*,* $\mathcal{T}$ *is an* $O(\sqrt{n})$*-approximate solution.*

(2) If G *is planar, then* $\mathcal{T}$ *is an* $O(\log n)$*-approximate solution.*

(3) If G *is an* H*-minor-free graph with* $|H| = h$*, then* $\mathcal{T}$ *is an* $O(h^2 \log n)$*-approximate solution.*

(4) If G *is a tree or an outerplanar graph, then* $\mathcal{T}$ *is an* $O(1)$*-approximate solution.*

In Section 4.1, we formulate the problem as a linear program. In Section 4.2, we describe how to apply the Whack-a-Mole algorithm [11] (with our own separation oracle) that obtains an $(1+\varepsilon)$-approximate solution to the linear program in polynomial time. In Section 4.3, we apply Chan and Har-Peled's randomized rounding technique [17], showing that bounding the pairwise product terms leads to the desired approximation factors. We show that there is a bound of $O(\sqrt{n})$ on any graph in Section 4.4, To bound the pairwise product terms, we introduce the scattering separators in Section 4.5. These scattering separators are useful for analyzing the approximation ratios for planar graphs in Section 4.6 and for minor-free graphs in Section 4.7. For the graph classes that is a subclass to the planar graphs, we provide tailored-but-better analysis for outerplanar graphs in Section 4.8 and for trees in Appendix B.8, which conclude the proof of Theorem 5.

4.1 LP Formulation

We formulate the c-balanced districting problem as a linear program. For each c-balanced star district S, we define a variable x_S indicating whether or not this district is chosen. Thus, the integer linear program for can be defined as:

$$\begin{aligned}
\text{maximize } & \sum_S w(S)x_S \\
\text{subject to } & \forall v \in V, \sum_{S \ni v} x_S \le 1 \\
& \forall S, x_S \in \{0,1\}
\end{aligned} \tag{2}$$

To give an approximate solution to the above integer linear program, we follow the standard recipe that solves the relaxed linear program first and then apply a randomized rounding algorithm.

Equivalent Relaxed Linear Program. In order to solve the relaxed linear program of Equation (2), we use *weighted* variables: for each district S, we define variable $x'_S := w(S)x_S$. Hence, the equivalent linear program (and its dual linear program) we will be solving can be described as follows.

(PRIMAL)	(DUAL)
maximize $\sum_S x'_S$	minimize $\sum_v y'_v$
subject to $\forall v \in V, \sum_{S \ni v} \frac{1}{w(S)} x'_S \le 1$	subject to $\forall S, \sum_{v \in S} \frac{1}{w(S)} y'_v \ge 1$
$x'_S \ge 0$	$y'_v \ge 0$

We note that the total number of primal variables (i.e., the number of potential c-balanced star districts) could be exponentially many in terms of the graph size. However, due to the special structure of this problem, seeking for an approximate solution does not require the participation of every variable. We summarize the result of solving the relaxed linear program as Theorem 6 below with details in Section 4.2 and Appendix B. A conclusive proof to Theorem 6 is carried out in Appendix B.5.

Theorem 6. *Given a graph G and a precision parameter $\varepsilon \in (0, \min\{\frac{1}{2}, \frac{c-2}{c}\})$, there exists an algorithm that returns an $(1-\varepsilon)$-approximate solution $\{x'_S, y'_v\}$ to the above linear program in* $\text{poly}(n, 1/\varepsilon, \log(w(G)))$ *time. Moreover, there are at most* $\text{poly}(n, 1/\varepsilon)$ *non-zero terms among the returned primal variables $\{x'_S\}$.*

4.2 Solving Relaxed Linear Program

We run a modified version of Whack-a-Mole Algorithm by Bhattacharya, Kiss, and Saranurak [11, Figure 3] to solve the dual linear program. In this section we only highlight the challenges and describe the modifications that resolve the challenges. For the sake of completeness, we provide a full walkthrough and the analysis of the algorithm in Appendix B.

At a high level glance, the Whack-a-Mole framework first reduces (via binary search on the objective value μ and re-scaling of the linear program) to the problem of reporting either a feasible primal solution of value ≥ 1 or a feasible dual solution of value $\le 1+\varepsilon$. To solve this reduced task, a multiplicative weights update (MWU) approach is applied: the algorithm initializes $y'_v = 1/n$, and iteratively finds any *strongly violating dual constraint*. A strongly violating dual constraint refers to a c-balanced district S such that $\sum_{v\in S} y'_v < (1-\varepsilon)\cdot w(S)$. Upon discovering any strongly violating constraint, the algorithm multiplicatively increases the weight of the vertices that are involved in the constraint. To guarantee an efficient runtime, the Whack-a-Mole algorithm uses a lazy and approximated MWU approach where it normalizes the dual variables $\{y'_v\}$ only when the sum exceeds $1+\varepsilon$. In the end, if all constraints are satisfied, then a feasible dual solution of value $\le 1+\varepsilon$ is found. Otherwise, the MWU framework guarantees a feasible primal solution after a sufficient number (namely, at least $\mu\varepsilon^{-2}\ln n$) of iterations.

The Challenges and the Solutions. The main obstacle from directly applying the BKS Whack-a-Mole algorithm is that their algorithm runs in near-linear time in terms of the number of non-zero entries. In particular, in their algorithm, every constraint is examined at least once. If a violating constraint is detected, a "whack" step is performed which corresponds to a MWU step and perhaps fixing that violation. In our case, the number of constraints of a dual linear program may be exponential in n. Thus, a straightforward analysis to a direct implementation of BKS algorithm leads to a running time exponential in n.

Fortunately, in our case it is not necessary to explicitly check every constraint. We develop a polynomial time *separation oracle* which either locates a good enough violating constraint or reports that all constraints are not strongly violating. With a tweaked potential method, we are able to show that, if in each iteration, we carefully pick a good enough violating constraint, the number of violating constraints being considered during the entire Whack-a-Mole process will be at most $\text{poly}(n, 1/\varepsilon)$. Therefore, by plugging in our separation oracle to the Whack-a-Mole algorithm, we are able to claim a polynomial runtime.

The Requirements to the Separation Oracle. Assume $\mu = 1$ (or we may assume that for all vertex v, the weights $w(v)$ are already replaced with $w(v)/\mu$) for simplicity. To obtain a polynomial upper bound on the number of iterations, our algorithm chooses some violating constraint that approximately maximizes its weight. Let us define the set of all *strongly violating cosntraints* as follows.

$$\mathcal{S}_{\text{violate}} := \left\{ S \;\middle|\; \sum_{v\in S} y'_v < (1-\varepsilon)\cdot w(S) \right\}. \tag{3}$$

Our separation oracle does the following: it either returns any (possibly weaker) violating S' such that

$$v \in S' \quad \text{and} \quad \sum_{v\in S'} y'_v < (1-\varepsilon/2)\cdot w(S') \quad \text{and} \quad w(S') \ge \frac{1}{2}\cdot \max_{T\in\mathcal{S}_{\text{violate}}} w(T), \tag{4}$$

or certifies that $\mathcal{S}_{\text{violate}} = \emptyset$.

Bounding the Number of Calls to the Separation Oracle. We show that the above modification leads to a polynomial upper bound the number of iterations. It suffices to bound the number of iterations within a single *phase* — within a phase, the dual variables are not normalized and are non-decreasing. At the beginning of a phase, the dual variables are normalized such that their sum becomes 1. Let us define a potential function ϕ for each vertex v. For brevity we denote $\mathcal{S}_v := \{S \in \mathcal{S}_{\text{violate}} \mid v \in S\}$.

$$\phi(v) := \begin{cases} 1 - \dfrac{y'_v}{\max\limits_{S\in\mathcal{S}_v} w(S)} & \text{if } \mathcal{S}_v \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Notice that, whenever there is a violating constraint S that contains v, it is guaranteed that $\phi(v) > \varepsilon$. This can be proved by contradiction: If $\phi(v) \le \varepsilon$, then for any $S \in \mathcal{S}_v$, we have

$$\sum_{u \in S} y'_u \ge y'_v \ge (1-\varepsilon) \cdot \max_{T \in \mathcal{S}_v} w(T) \ge (1-\varepsilon) \cdot w(S),$$

which implies that $S \notin \mathcal{S}_{\text{violate}}$ by Equation (3), a contradiction.

Suppose now a violating constraint S' is returned from the separation oracle and $v \in S'$. By Equation (4), we know that $\sum_{v \in S'} y'_v < (1-\varepsilon/2) \cdot w(S')$. After "whacking" the constraint, all vertices in S' have their dual value increased such that $\sum_{v \in S'} y'_v \ge w(S')$ (or the total sum $\sum_v y'_v$ becomes too large and a new phase is considered.) Let $\Delta y'_v$ denote the net increase of the value y'_v after "whacking" the constraint S'. Thus,

$$\sum_{v \in S'} \Delta y'_v \ge w(S') \cdot (\varepsilon/2), \tag{6}$$

which implies that, there exists at least one $v \in S'$ such that

$$\Delta y'_v \ge \frac{1}{|S'|} \cdot w(S') \cdot (\varepsilon/2) \ge \frac{1}{n} \cdot w(S') \cdot (\varepsilon/2) \ge \frac{\varepsilon}{4n} \cdot \max_{S \in \mathcal{S}_v} w(S). \tag{7}$$

We remark that the last inequality is due to the property of the violating constraint returned by the separation oracle (see Equation (4)). Now, we claim that the new potential at this particular vertex v must decrease by at least $\varepsilon/(4n)$ (or drop to 0). Indeed, we notice that after each whack, y'_v is non-decreasing and $\max_{S \in \mathcal{S}_v} w(S)$ is non-increasing (since the set of violating constraints $\mathcal{S}_v$ might shrink). When there are still strongly violating constraint after the update, the new potential function for v becomes:

$$\begin{aligned}
\phi_{\text{new}}(v) &= 1 - \frac{y'_v + \Delta y'_v}{\max\limits_{S \in \mathcal{S}_v^{\text{new}}} w(S)} \le 1 - \frac{y'_v + \Delta y'_v}{\max\limits_{S \in \mathcal{S}_v} w(S)} && (\{y'_u\} \text{ non-decreasing implies } \mathcal{S}_v^{\text{new}} \subseteq \mathcal{S}_v) \\
&= \left(1 - \frac{y'_v}{\max\limits_{S \in \mathcal{S}_v} w(S)}\right) - \frac{\Delta y'_v}{\max\limits_{S \in \mathcal{S}_v} w(S)} \le \phi(v) - \frac{\varepsilon}{4n}. && (\text{by (5) and (7)})
\end{aligned}$$

Therefore, we conclude that there can only be at most $O(n^2/\varepsilon)$ whacks during a phase, since there are n vertices and each can be whacked at most $O(n/\varepsilon)$ times. Finally, a bound of $O(\log n/\varepsilon^2)$ to the number of phases (see Lemma B.3) implies a desired polynomial upper bound.

Implementing the Separation Oracle. The last piece of the entire polynomial time algorithm would be to implement a desired separation oracle efficiently. The following lemma (Lemma 4.1) reduces the task of finding a violating district to solving the c-balanced star districting problem in a complete graph.

Lemma 4.1. *Given an input instance $G = (V, E, (p_1, p_2))$, re-weighted values $w' : V \to \{0\} \cup [\frac{1}{w(G)}, w(G)]$, and dual variables $y'_v \in [n^{-(1+1/\varepsilon)}, 1+\varepsilon]$ for each vertex $v \in V$, there exists an algorithm that either reports that $\mathcal{S}_{\text{violate}} = \emptyset$, or returns a c-balanced district S such that $\sum_{v \in S} y'_v < (1-\varepsilon/2)w'(S)$ and $w'(S) \ge \frac{1}{2}w'(S_{\max})$, where $S_{\max} = \arg\max_{S \in \mathcal{S}_{\text{violate}}} w'(S)$ is a violating c-balanced district with the maximum value. The algorithm runs in $O(\varepsilon^{-6} n^6 (\log n)(\log w(G))^4)$ time.*

Since the proof of Lemma 4.1 uses an FPTAS for the complete graph case (a more general version which looks for connected c-balanced districting is described in Section 5.1), we defer the proof to Appendix B.4. We remark that the proof of Lemma 4.1 is dedicated in the two-color and star districting setting. That said, if one would like to extend the framework to a more general setting (e.g., bounded diameter districting, general connected subgraph districting, or three-or-more colors setting), the implementation of the separation oracle has to be re-designed.

4.3 The Randomized Rounding Algorithm

We use a randomized rounding technique modified from [17, Section 4.3]. Intuitively, the algorithm maintains a set I of non-overlapping districts, which is initially empty, and keeps adding districts into I.

The rounding algorithm is described as follows. Let $\{x_S\}$ be the output of an approximate solution to LP. Let $\mathcal{S}_{\text{LP}} = \{S \mid x_S \neq 0\}$ be the support of the solution. The algorithm first sorts all non-zero valued districts according to their weights $w(S)$, from the largest to the smallest. Let $\tau \geq 1$ be a parameter to be decided later. For each district S, with probability x_S/τ, the algorithm adds S into I as long as there is no district in I overlapping with S. [1] The algorithm outputs I after all districts in $\mathcal{S}_{\text{LP}}$ are considered. The necessity of scaling the non-zero variables by τ comes from the analysis of expected total weight in I. In an actual implementation of the algorithm, one can make the algorithm oblivious of τ, by iteratively testing on different values of $\tau = (1+\varepsilon)^k$ for $k = 0, 1, 2, \ldots, O(\varepsilon^{-1} \log n)$ and then picking the largest weighted districting among the returned ones.

Analysis. The output I of the algorithm can be seen as a random variable. Let $w(I)$ be the total weight of the districts within I. A straightforward analysis (see Lemma B.4) shows that

$$\mathbf{E}[w(I)] \geq \sum_{S \in \mathcal{S}_{\text{LP}}} w(S) \frac{x_S}{\tau} - \sum_{A, B \in \mathcal{S}_{\text{LP}}:\ A \cap B \neq \emptyset} \min(w(A), w(B)) \frac{x_A x_B}{\tau^2} \ .$$

The right hand side of the above expression contains a weighted correlation term. The technique by Chan and Har-Peled [17] transforms the above weighted correlation terms into *unweighted* ones. They mentioned that, a desired $O(\tau)$-approximate solution can be achieved, as long as for any δ-*thresholded* subset $\mathcal{S}_{\geq\delta} := \{S \in \mathcal{S}_{\text{LP}} \mid w(S) \geq \delta\}$, the total unweighted correlation terms between overlapped districts can be upper bounded by the sum over all primal variables within the subset:

$$\sum_{A, B \in \mathcal{S}_{\geq\delta}: A \cap B \neq \emptyset} x_A x_B \leq \frac{\tau}{2} \cdot \sum_{S \in \mathcal{S}_{\geq\delta}} x_S \quad \forall \delta > 0 \tag{8}$$

The above condition implies the following (see Lemma B.5):

$$\mathbf{E}[w(I)] \geq \frac{1}{2\tau} \sum_{S \in \mathcal{S}_{\text{LP}}} w(S) \cdot x_S \geq \frac{1}{2\tau(1+\varepsilon)} \text{OPT}_{\text{LP}},$$

where OPT_{LP} is the optimal value of the LP problem. Thus, I is an $O(\tau)$-approximate solution in expectation. We remark that due to the factor 2 appearing in the denominator, this linear program with randomized rounding approach (although not contradicting) is unlikely to achieve a PTAS.

The above analysis to the rounding algorithm enables the approach of seeking a suitable τ value, such that Equation (8) holds. The rest of the section focuses on providing upper bounds of τ on various classes of input graphs.

[1] The randomization step appears to be necessary since there is an example where deterministic rounding incurs a large approximation factor, see Appendix B.10.3.

4.4 An $O(\sqrt{n})$-Approximation Analysis for General Graphs

In this section, we show that the algorithm achieves an $O(\sqrt{n})$-approximate ratio on any graph, by giving an upper bound $\tau = O(\sqrt{n})$ for the randomized rounding algorithm (with proof in Appendix B.9).

Lemma 4.2. *Let G be any graph. Let $\{x_S\}$ be any feasible solution to the linear program. Then,*

$$\sum_{A,B:\ A\cap B\neq\emptyset} x_A x_B \leq \sqrt{n}\cdot\sum_S x_S.$$

Remarks: Integrality Gap. We remark that this algorithm is achieving nearly the best approximation factor since it is NP-hard to have approximation factor of $n^{1/2-\delta}$ for any constant $\delta > 0$. Related to this, we would like to examine the potential loss in different steps of our algorithm. In the first step, we relax the integer linear program to a linear program with variables taking real numbers, the *integrality gap* refers to the ratio of the optimal fractional solution to the optimal integer solution (since we consider maximization problem the optimal fractional solution is no smaller than the optimal integer solution). In the second step of our algorithm, we use randomized rounding to turn the fractional solution back to a feasible integer solution. We call the ratio between the sum of products between overlapping districts' primal variables and the sum of all variables to be the *rounding gap*, i.e., $\sum_{A,B:A\cap B\neq\emptyset} x_A x_B / \sum_S x_S$. The rounding gap can be used to upper bound the loss of solution quality when we turn the fractional solution to a feasible integer solution using the randomized rounding algorithm.

Necessarily, a large integrality gap implies a large rounding gap for sure. Specifically, let τ be the rounding gap. The analysis to our rounding algorithm guarantees the existence of an integral solution within a factor of $4(1+\varepsilon)\tau$ from the optimal fractional solution, which implies an integrality gap of at most 4τ when setting $\varepsilon \to 0$. Thus if the integrality gap is large, we cannot have a small rounding gap. Interestingly, the above discussion, combined with Lemma 4.2 implies that the integrality gap of the natural LP formulation for the star districting problem is at most $O(\sqrt{n})$.

Next we show that our LP relaxation could have a large integrality gap of $\Omega(\sqrt{n})$ for a general graph. We use a reduction from k-uniform hypergraph matching problem to our c-balanced star districting problem. Let $H = (V_H, E_H)$ be the given k-uniform hypergraph – a hypergraph such that all its hyperedges have size k. We construct a graph $G = (V_H \cup E_H, E_G)$ by creating additional vertices for each hyperedge. These vertices have heavy weights, say $p_2(e) := (c-1)k$ and vertices from V_H have weights $p_1(v) := 1$. For each hypergraph $e \in E_H$ (which is a subset of vertices), we connect all vertices $v \in e$ to the corresponding vertex e in G. It ensures that there is an one-to-one correspondence between hyperedges of H to c-balanced star districts on G. Now, the relaxed linear program for (G, p_1, p_2) will be equivalent (with an extra ck-factor in the objective function) to a fractional hypergraph matching. Thus, the $(k + 1 - 1/k)$ integrality gap of k-uniform hypergraph matching [42, 18] can be transferred to our LP formulation — specifically, the construction in [18] via projective planes leads to an $\Omega(\sqrt{n})$ integrality gap.

On the other hand, we do observe *planar graph instances* with a constant > 1 rounding gap with an at most $1 + o(1)$ integrality gap (please see Appendix B.10). Again, this does not eliminate the possibility of achieving PTAS, but it suggests a conjecture that we are unlikely to obtain a PTAS for planar graphs using the current analysis.

4.5 Scattering Separators

Let us now introduce the scattering separators, which is useful for the divide and conquer framework for upper bounding the approximation ratio of the randomized rounding procedure.

Definition 4.3. Let $G = (V, E)$ be a graph and let $X \subseteq V$ be any subset. We say that X is:

- (k, t)-*scattered*, if X can be partitioned into at most k subsets $X = X_1 \cup X_2 \cup \cdots \cup X_k$ with each X_i being a t-hop independent set[2];
- (k, t)-*orderly-scattered*, if there exists a way to partition X into a sequence of at most k subsets $X = X_1 \cup X_2 \cup \cdots \cup X_k$, where each X_i is a t-hop independent set after the removal of all previous subsets $G - \cup_{j<i} X_j$.

Definition 4.4. Let G be a graph, $k, t \in \mathbb{N}$, and $\delta \in (0, 1)$. A (k, t, δ)-*scattering separator* is a subset of vertices $X \subseteq V$ such that (1) X is (k, t)-orderly-scattered, and (2) X is a balanced separator of G, that is, the largest connected component of $G - X$ has at most δn vertices.

We remark that a (k, t)-scattered set is also (k, t)-orderly-scattered. This orderly-scattered definition are useful when we remove subsets of vertices sequentially — they are used in the analysis of, for example, planar graphs and minor-free graphs. On the other hand, for some graph class such as outerplanar graphs it suffices to use (k, t)-scattered sets within the analysis.

The scattering separators are useful in the c-balanced districting problem for $t \geq 5$. To justify this, suppose that we have a 5-hop independent set Y. Any star district contains at most one vertex in Y. If two star districts contain different vertices of Y, the two districts must be disjoint. Thus we partition the pairs of overlapping districts by whether they overlap with Y or not, and if so, which vertex of Y. The following fact can be easily verified.

Fact 4.5. *Let Y be a 5-hop independent set. Consider a fixed district $A \in \mathcal{S}$. Assume there is a district $B \in \mathcal{S}$ that overlaps with A and $A \cup B$ touches Y, i.e., $A \cap B \neq \emptyset$ and $(A \cup B) \cap Y \neq \emptyset$. Since the diameter of $G[A \cup B]$ is at most 4, we know that $|(A \cup B) \cap Y| = 1$. Further, if A overlaps with two other star districts B, C with both centers of B, C in Y, then B, C have the same center.*

Fix a district $A \in \mathcal{S}$. Since all other districts that overlap with A contains (at most) the same vertex in Y, these primal variable values add up to at most 1. This implies that, removing Y from G charges at most one copy of $\sum x_S$. If we are able to show that the entire vertex set is a $(k, 5)$-orderly-scattered, then we obtain a desired $\tau = O(k)$ value for Equation (8). However, we do not know if such a constant k can be achieved for planar graphs. Fortunately, using the idea of balanced separators, we are able to achieve a polylogarithmic approximate solution.

Lemma 4.6. *If G and all its subgraphs have a $(k, 5, \delta)$-scattering separator, then the districting obtained from executing the algorithm on G is a $(2k \log_{1/\delta} n)$-approximated solution.*

Proof. Let $X = X_1 \cup X_2 \cup \cdots \cup X_k$ be a $(k, 5, \delta)$-scattering separator of G. Let $\mathcal{S}$ be the set of all districts. Then, all summands of the form $x_A x_B$ where $A, B \in \mathcal{S}$ and $A \cap B \neq \emptyset$ can be also split into three parts:

(1) $X \cap \{c_A, c_B\} \neq \emptyset$: one of the centers c_A or c_B is in X.

(2) $X \cap \{c_A, c_B\} = \emptyset$ but $X \cap A \cap B \neq \emptyset$: one of their common vertices is in X.

(3) None of the above.

For $j \in \{1, 2, 3\}$, we denote $cost_j$ the sum of products of those overlapping districts of case (j). For case (1), using the given constraint that X is $(k, 5)$-orderly-scattered, we consider removing each set X_i one at a time from the graph in the increasing order of i. For each X_i, without loss

[2] We say that X is a *t-hop independent set* (with respect to the graph G) if for all pairs of distinct vertices $u, v \in X$ and $u \neq v$, the shortest distance between u and v is at least t on G.

of generality, we may swap the role of A and B such that for each summand we have $c_B \in X$. By applying Fact 4.5 (with $Y = X_i$), we know that for each district $A \in \mathcal{S}$, all districts B that overlap with A with $c_B \in X_i$ are actually centered at the same vertex. This implies that the sum of all such x_B values will be at most 1 by the primal constraint. Hence, the contribution of any district $A \in \mathcal{S}$ under case (1) for X_i in the graph $G - \cup_{j<i} X_j$ is at most

$$\sum_{B: A\cap B \neq \emptyset \text{ and } c_B \in X_i} x_A x_B \leq x_A.$$

By summing over all $A \in \mathcal{S}$ and over all the k sets $X_1, \ldots, X_k$, we have $cost_1 \leq k \cdot (\sum_S x_S)$. For case (2), the terms can be partitioned according to the common vertex c:

$$cost_2 \leq \sum_{j=1}^{k} \sum_{c \in X_j} \sum_{c \in A \cap B} x_A x_B \leq \sum_{j=1}^{k} \sum_{c \in X_j} \left(\sum_{c \in A} x_A \right)^2 \leq \sum_{j=1}^{k} \sum_{c \in X_j} \sum_{c \in A} x_A \leq k \cdot \left(\sum_{S} x_S \right).$$

Again here we use the property that for any fixed vertex c, the sum of the primal variables for star districts containing c sum up to be at most 1, i.e., $\sum_{c \in A} x_A \leq 1$. Further, fix an X_i, any star district includes at most one vertex from X_i.

For case (3) we can delegate the cost to the recursion. Notice that, all districts whose centers are in X will not participate in case (3). Hence, when considering each of the connected component in $G - X$, all the districts (after chopping off vertices in X) are still connected and are star-shaped.

Since X is a balanced separator, the divide and conquer analysis has at most $\log_{1/\delta} n$ layers. Thus, the sum over all products of overlapping districts is bounded by at most $2k \log_{1/\delta} n$ times the sum $\sum_S x_S$. □

4.6 Planar Graphs

We are now able to derive an $O(\log n)$-approximation bound for planar graphs, immediately followed by the lemma below and Lemma 4.6.

Lemma 4.7. *Every planar graph has a* $(10, 5, 2/3)$*-scattering separator.*

The above separator lemma can be derived from the fundamental cycle separators, which is composed of two shortest paths on a BFS tree:

Fact 4.8 ([57])**.** *Let* $G = (V, E)$ *be a planar graph. Then, there exists a partition of* $V = L \cup X \cup R$*, such that (1) both* $|L|, |R| \leq \frac{2}{3}|V|$*, (2) no edges connect between* L *and* R*, and (3) the separator* X *is formed by the union of two root-to-node paths from some BFS tree on* G*.*

Proof of Lemma 4.7. With Fact 4.8, we know that there exists a separator X that is a union of two shortest paths $X = P_1 \cup P_2$. Since each P_i is a shortest path, we can partition each path into at most 5 sets and each set is a 5-hop independent set. Specifically, along a shortest path P_i, we color the vertices sequentially by color 1 to 5 in a round-robin manner. Each set of vertices of the same color is a 5-hop independent set. With two shortest paths, we have a total of 10 such 5-hop independent sets. Thus, each planar graph has a $(10, 5, 2/3)$-scattering separator. □

4.7 Minor-Free Graphs

A natural generalization of planar graph would be minor-free graphs. Let H be a graph with h vertices. Since planar graphs are $\{K_5, K_{3,3}\}$-free which implies K_6-free, one may expect the rounding algorithm achieves a similar approximation ratio bound for H-minor-free graphs as well.

Indeed, in this section, we show that our randomized rounding algorithm achieves an $O(h^2 \log n)$-approximation ratio for H-minor free graphs. Our proof is inspired by the seminal separator theorem of Alon, Seymour, and Thomas [2] for H-minor free graphs. But, our algorithm is substantially simpler since we do not require the separator to be small in size. Instead, all we need is to obtain a balanced separator X which is a union of at most $O(h^2)$ 5-hop independent sets. By removing these independent sets one after another, we are able to bound the correlation terms that involve any vertex in X by $O(h^2) \cdot \sum x_S$. Finally, an additional $\log n$ factor will then be added to the approximation ratio by the divide and conquer framework, achieving an $O(h^2 \log n)$ approximation ratio.

Since K_h-minor-free implies H-minor-free for any graph H of h vertices, the following separator theorem on K_h-minor-free graphs implies for all H-minor-free graphs.

Lemma 4.9. *Let $h \in \mathbb{N}$ be a constant. Let G be a K_h-minor-free graph. Then, G admits a $(5h^2, 5, 1/2)$-scattering separator.*

Proof. Consider an algorithm that *seeks for an K_h-minor*. The algorithm maintains two objects on the graph: a collection $\mathcal{C} := \{C_1, C_2, \ldots, C_k\}$ of disjoint vertex subsets, and an *active subgraph* B, such that:

1. For all i, the induced subgraph $G[C_i]$ is connected.
2. For all $i \neq j$, C_i and C_j are *neighboring*, that is, there exists an edge in G connecting some vertex from C_i and some vertex from C_j. If we contract every C_i into a vertex in $G[\cup_i C_i]$, we obtain a complete graph K_k. Hence, the algoritm maintains a certificate of K_k-minor[3] in G.
3. Let $X = \cup_i C_i$ be the separator of our interest. Whenever $|B| > n/2$, the active subgraph B is always the largest connected component[4] in $G - X$. We note that the largest connected component is uniquely defined whenever $|B| > n/2$.
4. Each C_i is $(5h, 5)$-orderly-scattered.

We give a high level description to our algorithm. The algorithm initially sets $\mathcal{C} = \emptyset$ and $B = G$. In each iteration, the algorithm repeatedly either finds a connected subgraph $C_{\text{new}} \subseteq V(B)$ that is neighboring to all $C_1, \ldots, C_k$, or removes some part C_i that is not neighboring to B. The algorithm halts once $|B| \leq n/2$.

Now we describe the detail of each iteration. If $|B| \leq n/2$ then we are done, as X is a $(5hk, 5, 1/2)$-scattering separator with $k \leq h$. Suppose that $|B| > n/2$. The algorithm would first check if there is a subset C_i that is not neighboring to B. If so, the algorithm removes C_i from $\mathcal{C}$. The invariants 1, 2, and 4 clearly holds for $\mathcal{C} - C_i$ so it suffices to show that the third invariant holds as well. Since B has no neighbor in C_i and B is a connected component, we know that B is still a connected component of $G - (\cup_{j \neq i} C_j)$. The fact that $|B| > n/2$ implies that all connected components other than B has at most $n/2$ vertices, so the third invariant holds.

Now we can assume that all subsets $C_1, C_2, \ldots, C_k$ in $\mathcal{C}$ have neighboring vertices in B. In this case, we claim that a new set $C_{\text{new}} \subseteq V(B)$ can be found such that we update $\mathcal{C} \leftarrow \mathcal{C} \cup \{C_{\text{new}}\}$ and that the invariant 1, 2, and 4 holds for the updated $\mathcal{C}$. To construct C_{new}, we first let $a_1, a_2, \ldots, a_k$ be vertices in B such that for all i, a_i has a neighbor in C_i. Since B is a connected component, there exist paths connecting a_i to a_{i+1} in B, for all i, $1 \leq i \leq k-1$. Let P_i be the *shortest path* from a_i to a_{i+1} in B. We define $C_{\text{new}} := \cup_{i=1}^{k-1} V(P_i)$. Invariants 1 and 2 are clearly satisfied for C_{new}. The fact that P_i is a shortest path implies that $V(P_i)$ can be partitioned into at most 5

[3] In [2], they called $\mathcal{C}$ a *covey*.

[4] In [2], they called each connected component in $G - X$ an X-flap.

5-hop independent sets. Thus, C_{new} can be partitioned into at most $5h$ 5-hop independent sets, thereby maintaining invariant 4. Since G is K_h-minor-free, we know that $|\mathcal{C}| \le h-1$.

After constructing the set C_{new} and updating $\mathcal{C} \leftarrow \mathcal{C} \cup \{C_{\text{new}}\}$, the algorithm also updates the active subgraph B, by substituting B with the largest connected component B' of $G - \cup_{C_i \in \mathcal{C}} C_i$. Notice that, if $|B'| \le n/2$ then $X = \cup_{C_i \in \mathcal{C}} C_i$ is already a balanced separator. Otherwise, the fact that B is the unique connected component with $|B| > n/2$ implies that $B' \subseteq B - C_{\text{new}}$. Hence, in any case we have $|B'| < |B|$. That is, the active subgraph strictly decreases in size. The algorithm must halt, since in each iteration the quantity $|\mathcal{C}| + 2|B|$ is nonnegative but strictly decreasing – either we drop a set in $\mathcal{C}$, thus $|\mathcal{C}|$ drops but $|B|$ remains the same, or we increase $|\mathcal{C}|$ by 1 but decrease $|B|$ by 1 at least. Therefore, whenever the algorithm halts, we obtain a separator $X = \cup_i C_i$ as desired, which implies Lemma 4.9. □

Remarks. If we apply Lemma 4.9 to planar graphs, we may use the fact that planar graphs are K_6-minor-free and thus obtaining a $(180, 5, 1/2)$-scattering separator. This leads to an $(360 \log_2 n)$-approximation ratio by applying Lemma 4.6. Note that this is much worse than $10 \log_{3/2} n \approx 17.1 \log_2 n$, which is the bound we obtained from Lemma 4.7.

4.8 Outerplanar Graphs

The results from previous subsections suggest approximation ratios that involve polylogarithmic terms. It is, of course not surprising, possible to obtain a better approximate ratio for subclasses of planar graphs. In this subsection, we show that our algorithm produces an $O(1)$-approximate solution for outerplanar graphs, using a slightly different approach.

Lemma 4.10. *Let G be an outerplanar graph. Suppose that $\{x_S\}$ are primal variables obtained by Theorem 6. Then, $\sum_{A \cap B \neq \emptyset} x_A x_B \le O(1) \cdot \sum x_S$.*

We defer the proof of Lemma 4.10 to Appendix B.7. For an even simpler graph classes such as trees, we are able to obtain a much smaller constant bound. We provide such an analysis in Appendix B.8.

5 FPTAS for General Districting on Complete Graphs and Trees

In this section, we present FPTAS for complete graphs and trees with weighted blocks. The algorithms here find c-balanced, connected districts that can be more than a star graph. Further, for complete graphs and trees, the LP-based algorithm in the previous section achieves $O(1)$-approximation ratio while the algorithms in this section achieves a ratio of $1 + \varepsilon$.

5.1 Complete Graph

Let G be a complete graph with functions of weights p_1 and p_2. Because we can merge two adjacent c-balanced districts on G into a single c-balanced district as shown in Fact 5.1, the c-balanced districting problem on complete graphs can be reduced to obtaining *one* c-balanced district, described as the following:

Fact 5.1 (Mergeable Property)**.** *Assume T_1 and T_2 are disjoint districts and $G[T_1 \cup T_2]$ is connected. If T_1 and T_2 are both c-balanced, then $T_1 \cup T_2$ is also a c-balanced district.*

Complete-Graph-c-Balanced-Districting
Input: Let $G = (V, E)$ be a complete graph of n blocks and function of weights p_1 and p_2.
Goal: Obtaining a subset $S \subseteq V$ such that the total weight $w(S)$ is maximized subjected to the c-balanced condition:

$$(c-1)p_1(S) - p_2(S) \geq 0 \qquad \text{and} \qquad (c-1)p_2(S) - p_1(S) \geq 0. \tag{9}$$

The following theorem gives an FPTAS using dynamic programming (Algorithm 1).

Algorithm 1: FPTAS on complete graphs

```
Input: ε > 0, c > 2, a complete graph (V, E), V = {v_1, ..., v_n}, functions of weights
       p = (p_1, p_2)
Function Trim(L, ℓ, ε):
    Sort L = {q_1, ..., q_m} so that ℓ(q_1) ≥ ℓ(q_2) ≥ ··· ≥ ℓ(q_m);
    Set L_out = ∅;
    for i = 1, ..., m do
        if q_i is not marked then
            L_out ← L_out ∪ {q_i};
            Mark all q_j ∈ L that ε-approximates q_i;
    return L_out;
Set L_1^0 = L_2^0 = {(0, 0)};
for i = 1, ..., n do
    L_1^i ← Trim(L_1^{i-1} ∪ (L_1^{i-1} + p(v_i)), ℓ_1, ε/n);
    L_2^i ← Trim(L_2^{i-1} ∪ (L_2^{i-1} + p(v_i)), ℓ_2, ε/n);
return the largest c-balanced districting in L_1^n ∪ L_2^n;
```

Theorem 7. *There exists an FPTAS algorithm solving* Complete-Graph-c-Balanced-Districting *so that for all $c > 2$, $0 < \varepsilon < \frac{1}{2}\ln(c-1)$, and complete graph (V, E) of n nodes with functions of weights $\mathbf{p} = (p_1, p_2)$, the algorithm outputs an e^ε-approximation in $O\left(\varepsilon^{-4} n^6 (\ln w(V))^4\right)$ time where $w(V) = \sum_{v \in V} p_1(v) + p_2(v)$.*

We defer a detailed proof of Theorem 7 to Appendix C and here give a high level idea. One naive approach involves creating a complete list of potential subset sum values, denoted as $L(V)$ and outputting the largest c-balanced one. While this approach finds an optimal solution, it is not necessarily efficient, as $L(V)$ can be exponentially large. Similar to the knapsack problem or subset sum problem, one may use a bucketing idea to trim the list, keeping only one value when several are close to each other. However, the c-balanced constraint posts a challenge for the algorithm — for example, if an trimming algorithm keeps partial districts during the iterations, these partial districts may not always remain c-balanced resulting in a poor approximation ratio. To address this, we design a prioritized trimming process that prioritizes subsets satisfying the c-balanced condition in Equation (9) such that any c-balanced district in $L(V)$ would have an approximated district in our trimmed list.

Specifically, given $\varepsilon \geq 0$, we say $\mathbf{q}$ is an *ε-approximate* of $\mathbf{q}'$ if $q_1/q_1', q_2/q_2' \in [e^{-\varepsilon}, e^{\varepsilon}]$ where $0/0 := 1$. Let $\ell_1(\mathbf{q}) = (c-1)q_1 - q_2$ and $\ell_2(\mathbf{q}) = (c-1)q_2 - q_1$ be two linear functions on $\mathbf{q} = (q_1, q_2) \in \mathbb{R}^2$. We say $\mathbf{q}$ is *ℓ_j-dominated* by $\mathbf{q}'$ if $\ell_j(\mathbf{q}) \leq \ell_j(\mathbf{q}')$ for $j = 1, 2$, and $\mathbf{q}, \mathbf{q}' \in \mathbb{R}^2$, and L' is a *(ℓ_j, ε)-trimmed* of L if $L' \subseteq L$ and for each $\mathbf{q} \in L$ there exists $\mathbf{q}' \in L'$ which ε-approximates and ℓ_j-dominates $\mathbf{q}$. The key observation is that if $\mathbf{q}$ is c-balanced satisfying Equation (9) with

$q_2 \geq q_1$ and $\mathbf{q}'$ ℓ_1-dominates $\mathbf{q}$, $\mathbf{q}'$ is also c-balanced. A similar argument holds for $q_1 \geq q_2$. This observation suggests that when trimming multiple nearby values, we keep the one that optimizes ℓ_1 (and ℓ_2) that ensures the existence of c-balanced approximated values. Therefore, we can find an e^{ε}-approximated solution if we can compute (ℓ_j, ε)-trimmed of all possible subset sum values $L(V)$. Moreover, because ℓ_1 and ℓ_2 are linear, we can use dynamic programming to sequentially and efficiently compute L_1^i and L_2^i that is $(\ell_1, \frac{\varepsilon i}{n})$-trimmed and $(\ell_2, \frac{\varepsilon i}{n})$-trimmed of all possible subset sum values on the first i blocks $L^i = L(\{v_1, \dots, v_i\})$ respectively. While Algorithm 1 only returns the size of our approximated solution $\mathbf{q} \in \mathbb{R}^2$, we can use an additional n factor to store the set S for each $\mathbf{q}$ in L_1^i, L_2^i to recover our approximated optimal districting.

Finally, we note that our prioritized trimming that ensures both inequalities in Equation (9): one through prioritized ℓ_j the other through exhausting cases of $q_2^* \geq q_1^*$ or $q_2^* \leq q_1^*$. However, we cannot extend this approach to non-binary color settings. Instead, if we allow relaxing c-balanced constraint to c'-balanced district with c' slightly larger than c, the standard bucketing algorithm mentioned above can directly work even for the non-binary color setting.

5.2 Tree Graph

Similar to our dynamic programming algorithm for complete graphs, we can design an FPTAS for trees. First, note that if we only need to find one c-balanced district, we can easily adapt our FPTAS for a complete graph. Concretely, we recursively grow (incomplete) districts for a block's children in a depth-first search (DFS) order and decide whether to continue growing the district or not. For more than one district, we can use additional memory to store the total weight of all c-balanced districts that are fully contained in the subtree of each block and provide an approximate districting solution as described in Algorithm 5.

Theorem 8. *There exists an FPTAS for the c-balanced districting problem on trees. That is, for all $c > 2$, $0 < \varepsilon < \frac{1}{2}\ln(c-1)$, and tree graph (V, E) of n blocks with population functions $\mathbf{p}$, the algorithm outputs an e^{ε}-approximation in $O\left(\varepsilon^{-6} n^8 (\ln w(V))^6\right)$ time.*

We provide a detailed proof of Theorem 8 in Appendix C and provide the intuition of proof here. We need additional notations for our algorithm. Given a districting $\mathcal{T}$ on a rooted tree (V, E), we define a sequence of *stamps* $\mathbf{s}^v(\mathcal{T}) = (s_1^v(\mathcal{T}), s_2^v(\mathcal{T}), s_3^v(\mathcal{T})) \in \mathbb{R}^3$ for $v \in V$ where the first two coordinates store information about the current incomplete district; and the third coordinate is the total weight of completed c-balanced districts in the subtree of v

$$s_3^v(\mathcal{T}) = \sum_{T \in \mathcal{T}: T \text{ is in the sub-tree of } v} w(T).$$

We define $s_1^v(\mathcal{T})$ and $s_2^v(\mathcal{T})$ according to the following three cases.

1. If v is not in any district of $\mathcal{T}$, we call v *absent* and set $s_1^v = s_2^v = 0$.

2. If there exists a district $T \in \mathcal{T}$ that is fully contained in the subtree of v and $v \in T$, we call v is *consolidating* and set $s_1^v = s_2^v = 0$.

3. Finally, if $T \in \mathcal{T}$ is not fully contained in the subtree of v and $v \in T$, we call v *incomplete* (for T or $\mathcal{T}$). Additionally, we let $T^v \subsetneq T$ be the subtree of T consisting of v's descendant including v, and set $s_1^v = p_1(T^v)$ and $s_2^v = p_2(T^v)$ which are the population of each community in the incomplete district T^v.

Note that given $\mathcal{T}$ and v, if v is incomplete for $T \in \mathcal{T}$, $s_1^v(\mathcal{T}) = s_1^v(\{T\}) = p_1(T^v)$ and $s_2^v(\mathcal{T}) = s_2^v(\{T\}) = p_2(T^v)$.

Our FPTAS (Algorithm 5) uses a prioritized trimming process similar to Algorithm 1 and keeps stamps of all possible c-balanced districtings in the depth-first search (DFS) order:

$$L^v = \{\mathbf{s}^v(\mathcal{T}) : \mathcal{T} \text{ is a } c\text{-balanced districting}\}. \tag{10}$$

Similarly, while Algorithm 5 only returns the size of our approximated solution, we can use an additional n factor to store one $\mathcal{T}$ for each $\mathbf{s}$ in L_1^v, L_2^v to recover our approximated optimal districting.

Adapting to the Star Districting Setting. We note that this dynamic programming approach also works for star districts on tree graph and yields an FPTAS. Similar to the arbitrary districts setting, we consider three cases for each v: the absent case where v is not included in any district; the consolidating where v is in a star district that is contained in its descendants; the incomplete case where v is the center of a star district that is incomplete.

6 Greedy Algorithms for Special Settings

In this section we discuss greedy strategies for various special cases of the c-balanced districting problem.

6.1 General graphs with bounded rank districts

Given $G = (V, E)$ with functions of weights p_1 and p_2 and integer k, the problem of c-balanced districting with rank k asks for a districting $\mathcal{T}$ such that each $T \in \mathcal{T}$ is a c-balanced district and $|T| \leq k$. Further, the total weight $w(\mathcal{T}) := \sum_{T\in\mathcal{T}}(p_1(T) + p_2(T))$ should be maximized.

We first provide a reduction from the most general c-balanced districting problem to maximum weighted matching on hypergraphs. In a hypergraph $H = (V_H, E_H)$, $V_H = V$ and each hyperedge $h \in E_H$ is a subset of V. The *rank* of h is defined as $|h|$ and the rank of H is the maximum rank among all $h \in E_H$. If all hyperedges have the same rank r, we say that H is *r-uniform*. A hypergraph matching $M \subseteq E_H$ on H is a collection of pairwise disjoint hyperedges. For a given weight function $w : E_H \to \mathbb{Z}_{\geq 0}$, the maximum weighted hypergraph matching problem seeks for a matching M that maximizes $w(M) := \sum_{h\in M} w(h)$.

Reduction. The reduction is straightforward: we treat c-balanced districts on G as hyperedges in H. Specifically, for each c-balanced district T with $|T| \leq k$, we create a hyperedge $h = T$ and add h to the hypergraph. We can even enforce h to have rank-k, by adding dummy vertices to the hypergraph and h. The weight $w(h)$ is then defined naturally as the sum of vertex weights within h, namely $\sum_{v\in h} w(v)$. The dummy vertices have weight 0.

Now, it is clear that any matching M in H has a one-to-one correspondence to a valid bounded-rank-k districting $\mathcal{T}$ on G. Therefore, any algorithm solving the weighted hypergraph matching problem on a k-uniform hypergraph H can be applied to solving the c-balanced districting problem with bounded rank k.

Simple Case: $k = 2$. The problem become polynomial time solvable by regarding the weighted hypergraph matching problem as a maximum weighted matching problem on normal graphs. We summarize the result below.

Theorem 9. *There exists an algorithm that given an instance of c-balanced districting problem with rank $k = 2$ on $G = (V, E)$ with population functions p_1 and p_2 outputs the solution in polynomial time.*

Proof. Following the reduction, first consider the 2-uniform hypergraph (V_H, E_H) formed using valid districts as hyperedges. $|V_H| = n$ and $|E_H| = O(n^2)$. Since the hypergraph is 2-uniform, the problem now reduces to maximum weight matching on general graphs. The fact that H can be formed in linear time and we can solve maximum weight matching in $O(|E_H|\sqrt{|V_H|}\log(|V_H| \cdot w(G)))$ time [38, 43] completes the proof. □

General Case: $k > 2$. For c-balanced districting with rank k for any k, we give a greedy algorithm that achieves a k-approximation ratio. Consider the greedy strategy described below in Algorithm 2.

Algorithm 2: A Polynomial Time Greedy Algorithm for Hypergraph Matching

```
Input: Given H = (V_H, E_H) with the weight function w.
Initialize an empty matching M = ∅;
while E_H ≠ ∅ do
    Pick h ∈ E_H with the largest weight (breaking ties arbitrarily) and add h to M.
    Remove all edges h' with h' ∩ h ≠ ∅ from E_H.
return M
```

Lemma 6.1. *For a given hypergraph $H = (V_H, E_H)$ with rank at most k, Algorithm 2 returns a k-approximate solution to the maximum weight hypergraph matching problem.*

Proof. Given (V_H, E_H), let M^* be the optimal solution, and M be the output of Algorithm 2. Consider the mapping $f : M^* \to M$: for each $h \in M^*$, $f(h)$ gives the edge with the largest weight that shares at least one vertex with h, i.e.

$$f(h) = \operatorname*{argmax}_{h' \in M : h' \cap h \neq \emptyset} w(h').$$

Note that the set $\{h' \cap h \neq \emptyset\}$ is nonempty and thus f is well-defined: otherwise, h itself must be added to M by the greedy algorithm. Furthermore, since the greedy algorithm considers districts with a larger weight first, we have $w(h) \leq w(f(h))$. Finally, because all edges in M^* have disjoint sets of vertices, each $h' \in M$ has at most k edges in the preimage $f^{-1}(h')$. Thus,

$$w(M^*) = \sum_{h \in M^*} w(h) \leq \sum_{h \in M^*} w(f(h)) \leq k \sum_{h' \in M} w(h') = k \cdot w(M).$$

Therefore, M is a k-approximate solution. □

Using Lemma 6.1, we obtain an approximation algorithm for our c-balanced districting with rank k.

Theorem 10. *There exists an algorithm that given an instance of c-balanced districting problem with rank k for $k \geq 3$ on G with functions of weights p_1 and p_2 outputs a k-approximate solution in $\tilde{O}(kn^k)$ time.*

Proof. Consider the rank k hypergraph H obtained from G such that each hyperedge in H corresponds to a c-balanced district. Following the reduction and Lemma 6.1, we have that Algorithm 2 returns a k-approximate solution to the c-balanced districting problem with rank k. The set H can be constructed in time $O(n^k)$, sorting and running the greedy algorithm takes additional $\tilde{O}(kn^k)$ time. □

6.2 General graphs with bounded degree and star districts

In this section we consider c-balanced districting problem on a graph G with arbitrary weights that has maximum degree bounded by Δ. Since each c-balanced district on G now has at most $\Delta+1$ blocks, by Theorem 10 we obtain an algorithm that produces a $(\Delta+1)$-approximate solution in $\tilde{O}(n\Delta 2^{\Delta})$ time.

A Slightly Improved Approximation Ratio. We realized that, using a special property of star districts, we are able to achieve a (very slightly) improvement to the approximation ratio, from $\Delta+1$ to $\Delta+\frac{1}{\Delta}$. The observation is as follows. Suppose that a tight $(\Delta+1)$-approximated solution is returned from the greedy algorithm (via Algorithm 2). Let T be one of the returned district with $|T| = \Delta+1$. Then, because $w(T)$ is charged for $\Delta+1$ times in the analysis, every vertex of T belongs to *distinct* districts in an optimal solution. In the star districting setting, we know that the center vertex c_T must form a solo c-balanced district and occur in the optimal solution.

The above observation leads to a local-swap idea: if a sufficiently heavy center vertex c_T of a district T that is about to be added to the solution by the greedy algorithm, we can simply add $\{c_T\}$ to the output instead of adding T (which potentially blocks lots of other districts). We implement this idea in Algorithm 3, where the only difference is highlighted as "special case".

Algorithm 3: Greedy algorithm for star districts on bounded-degree graphs

```
Input: Input graph G = (V, E) with maximum degree Δ, c ≥ 2.
Output: A collection of c-balanced star districts 𝒯 = {T_1, ..., T_ℓ}
Initialize 𝒯 = ∅, 𝒮 = ∅.
for v ∈ V do
  | Add all c-balanced star districts centered at v (by enumeration) to 𝒮.
Sort districts in 𝒮 by their weights in the decreasing order.
while 𝒮 is not empty do
  | Let T_i be the largest weight district in 𝒮 and let c_i be the center of T_i.
  | if {c_i} is balanced and |T_i| = Δ + 1 and w(c_i) ≥ w(T_i)/Δ then // special case
  |   | Add {c_i} to 𝒯.
  |   | Remove all districts that contains c_i from 𝒮.
  | else
  |   | Add T_i to 𝒯.
  |   | Remove all districts that overlap with T_i from 𝒮.
return 𝒯
```

Theorem 11. *There exists an algorithm that given a graph $G = (V, E)$ with maximum degree bounded by Δ and population functions $p_i(v) \in \mathbb{Z}_{\geq 0}$ for $v \in V$ outputs a $(\Delta + \frac{1}{\Delta})$-approximate solution to the c-balanced districting problem with star districts in $\widetilde{O}(n\Delta 2^{\Delta})$ time.*

Proof. The idea is similar to proof of Lemma 6.1. Let $\mathcal{T}^* = \{T_1^*, \cdots, T_{\ell^*}^*\}$ be any optimal districting and $\mathcal{T}$ be the districting returned by Algorithm 3. Let $f : \mathcal{T}^* \to \mathcal{T}$ be the function that for $T_i^* \in \mathcal{T}^*$ returns a district in $\mathcal{T}$ with largest weight that intersects with T_i^* in at least one vertex. If the special case does not occur, we know that $w(T_i^*) \leq w(f(T_i^*))$. Otherwise, we know by the condition of the special case, we have $w(T_i^*) \leq \Delta \cdot w(f(T_i^*))$. A naive bound gives $\sum_{T_i^* \in \mathcal{T}^*} w(T_i^*) \leq \sum_{T\in\mathcal{T}} w(T)|f^{-1}(T)|$ – in the worst case $|f^{-1}(T)|$ can be as large as $\Delta+1$.

Now, to provide a better approximation ratio bound, it suffices for us to bound for each $T \in \mathcal{T}$, how many times $w(T)$ is charged by the districts in the optimal solution. We say that T is *triggered* by the special case, if T is a solo c-balanced district that joins $\mathcal{T}$ according to the special case in Algorithm 3. There are three cases:

- **Case 1:** $|f^{-1}(T)| \le \Delta$ and T is not triggered by the special case. In this case, the contribution of T to the sum would be $w(T) \cdot |f^{-1}(T)| \le \Delta \cdot w(T)$.
- **Case 2:** $|f^{-1}(T)| = 1$ and $T = \{c_i\}$ is triggered by the special case. In this case, when T joins the output set $\mathcal{T}$, the district that the algorithm is actually considering has weight at most $\Delta \cdot w(T)$. Hence, the district in the optimal solution $T^* \in f^{-1}(T)$ must satisfy $w(T^*) \le \Delta \cdot w(T)$.
- **Case 3:** $|f^{-1}(T)| = \Delta+1$ and T has not been triggered by the special case. In this case, there are exactly $\Delta+1$ districts in the optimal solution. Using the observation, the center vertex c_T forms a solo district in the optimal solution. Moreover, since the special case is not triggered for T, we know that $w(c_T) \le w(T)/\Delta$. Therefore, $\sum_{T^* \in f^{-1}(T)} w(T^*) \le (\Delta + \frac{1}{\Delta})w(T)$ as desired.

Since in any case the contribution of $w(T)$ in the anaylsis does not exceed $(\Delta+\frac{1}{\Delta})w(T)$, we conclude that Algorithm 3 produces an $(\Delta + \frac{1}{\Delta})$-approximate solution.

For runtime, note that the c-balanced district for each node can be found in time $O(n2^\Delta)$ and the sorting and greedy strategy can be implemented in time $\widetilde{O}(n\Delta 2^\Delta)$. □

Remark. The exponential dependence on Δ in runtime can be reduced to $\text{poly}(\Delta, 1/\varepsilon)$ at the cost of an additional $(1+\varepsilon)$ term in the approximation factor. This can be achieved by using the FPTAS for complete graph (Algorithm 1) for finding the largest c-balanced district centered at each vertex v in Algorithm 3.

Binary Weights. If we impose an additional constraint such that the input graph G has a bounded degree Δ and each vertices have either p_1 or p_2 weight to be 1, we may achieve a $(\Delta+1)/2$-approximate solution. Consider an algorithm that computes any maximum (cardinality) matching M^* where for each matched edge $\{u, v\} \in M^*$ we have $p_1(u) = 1$ and $p_2(v) = 1$. The algorithm simply treats each matched edge as a perfectly balanced district and returns $\mathcal{T}_{\text{ALG}} = M^*$. Clearly, $w(\mathcal{T}_{\text{ALG}}) = 2|M^*|$.

To see the approximation ratio, consider any optimal solution $\mathcal{T}^*$. Each district $T \in \mathcal{T}^*$ contains at least one vertex with p_1 weight 1 and one vertex with p_2 weight 1. Thus, $|M^*| \ge |\mathcal{T}^*|$. On the other hand, since each star district $T \in \mathcal{T}^*$ has at most $\Delta + 1$ blocks, we conclude that

$$w(\mathcal{T}^*) \le (\Delta+1)|\mathcal{T}^*| \le (\Delta+1)|M^*| = \frac{\Delta+1}{2} w(\mathcal{T}_{\text{ALG}}).$$

Since we are pairing up vertices of different weight types, we are actually solving for a maximum unweighted bipartite graph matching. The state-of-the-art unweighted maximum bipartite matching can be solved in $m^{1+o(1)}$ time [19], $\tilde{O}(m + n^{1.5})$ time [66], or $n^{2+o(1)}$ time [10, 26].

6.3 General graphs with binary weights and star districts

For binary weight setting, we assume that each vertex v has only one type of non-zero population/weight, i.e. either $p_1(v) = 1$ or $p_2(v) = 1$. We call a vertex *covered* if it is part of some districting $T \in \mathcal{T}$ and *uncovered* otherwise.

We use local search to find an approximate solution, with details in Algorithm 4. We prove that this algorithm returns a c-approximate districting. The idea is to argue that for *any* optimal district $T_i^* \in \mathcal{T}^*$, at least c fraction of nodes in T_i^* are covered by some $T_i \in \mathcal{T}$, the approximation guarantee then follows. We do this on a case basis, by considering a fixed T_i^* and for its center vertex, based on the membership in districting $\mathcal{T}$ argue for the covering of its neighbor vertices.

Algorithm 4: Local search algorithm for star districts on graphs with binary weights

Input: Input graph $G = (V, E)$ with binary weights, $c \geq 2$.
Output: Balanced star districts $\mathcal{T} = \{T_1, \cdots, T_\ell\}$
Initialize $\mathcal{T} = \emptyset$, k to be the largest integer $\leq c - 1$.
while $\exists (u, w) \in E$ *such that* $p_1(u) = 1$, $p_2(w) = 1$ *and* u *and* w *are uncovered* **do**
  Start a new district T, setting any one of the nodes as the center (assume w)
  Greedily assign uncovered neighbors of w to T while maintaining c-balanced property of T
  for $v \in T \setminus \{w\}$ **do**
    if v *has* $> k$ *uncovered neighbors such that* v *forms a* c*-balanced district with these vertices* **then**
      Remove v from T and start new district T' with v as center; `// swap step`
      Assign the $> k$ uncovered neighbors to v.
  Add all such created T' to $\mathcal{T}$.
  If the resulting T after swap steps is still balanced, add T to $\mathcal{T}$.
Output $\mathcal{T} = \{T_1, \cdots, T_\ell\}$

Theorem 12. *The districting $\mathcal{T}$ returned by Algorithm 4 on termination is a c-approximate solution.*

Proof. Let $\mathcal{T}$ denote the set of districts returned by Algorithm 4 and $\mathcal{T}^*$ be the set of optimal districts. Consider $T_i^* \in \mathcal{T}^*$ in the optimal solution and let c_i be its center, $\{v_1, \cdots, v_m\}$ be the children of c_i in T_i^* and assume wlog $p_1(c_i) = 1$.

Case 1: Suppose $c_i \notin T_i$ for all $T_i \in \mathcal{T}$. Since we have $p_1(c_i) = 1$, the subset of vertices in $\{v_1, \cdots, v_m\}$ that are not covered by any district in $\mathcal{T}$ have p_1 weight to be 1 (and p_2 weight of 0), as otherwise the algorithm would use the edge of any of those vertices to c_i to start a new district. This implies all the vertices with p_2 weight of 1 in $\{v_1, \cdots, v_m\}$ are covered in $\mathcal{T}$ which gives us that at least $|T_i^*|/c$ vertices are covered.

Case 2: Suppose that c_i is also a center of $T_i \in \mathcal{T}$. Assume wlog that majority of vertices in T_i have p_1 weight of 1. Since the algorithm terminates, it means that no additional vertex with p_1 weight 1 from $\{v_1, \cdots, v_m\}$ can be added without violating the c-balanced property. It also implies that vertices with p_2 weight 1 from $\{v_1, \cdots, v_m\}$ are covered in $\mathcal{T}$ (otherwise the algorithm could just include those vertices since T_i is has majority of vertices weight p_1 weight of 1, giving us that at least $|T_i^*|/c$ vertices are covered. The case where T_i has majority vertices with p_2 weight 1 follows via similar argument.

Case 3: Suppose $c_i \in T_i$ but it is not the center. We have the following two cases which follow from the conditions on the "swap" step within the algorithm:

- **Case (3.a):** The swap step could not create a new district with c_i as the center. Recall we assume with loss of generality that $p_1(c_i) = 1$. Since a new district could not be created, neighbors of c_i with p_2 weight 1 (which also include vertices from $\{v_1, \cdots v_m\}$ in T_i^*) were already covered by some T. This in turn implies that at least $|T_i^*|/c$ vertices are covered.

- **Case (3.b):** Creating a new district with c_i was possible but the improvement was not substantial. This case occurs when the district resulting from swapping c_i as center results in at most k additional vertices being covered. As creating a new district was possible, we have that c_i has at least one uncovered neighbor with p_2 weight 1. This implies at least $c-1$ vertices with either p_i weight could be added to this potential district without violating the c-balanced property, but were not available. As c_i is not a center from swap step, at most k vertices from $\{v_1, \cdots, v_m\}$ are left uncovered by $\mathcal{T}$. If $|T_i^*| < c$, since c_i is covered, it follows that at least $1/c$ fraction of vertices are covered. Now, for $|T_i^*| \geq c$, $k \leq c-1$ and $c \geq 2$ we have $|T_i^*| - (c-1) \geq \frac{|T_i^*|}{c}$ which gives us the desired approximation factor.

□

References

[1] M. Allman, I. Ashlagi, I. Lo, J. Love, K. Mentzer, L. Ruiz-Setz, and H. O'Connell. Designing school choice for diversity in the San Francisco unified school district. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, page 290–291, New York, NY, USA, 2022. Association for Computing Machinery.

[2] N. Alon, P. D. Seymour, and R. Thomas. A separator theorem for graphs with an excluded minor and its applications. In H. Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 293–299. ACM, 1990.

[3] M. Altman. Traditional districting principles: judicial myths vs. reality. *Social Science History*, 22(2):159–200, 1998.

[4] M. Altman and M. P. McDonald. Bard: Better automated redistricting. *Journal of Statistical Software*, 42:1–28, 2011.

[5] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

[6] P. Austrin, S. Khot, and M. Safra. Inapproximability of vertex cover and independent set in bounded degree graphs. *Theory Comput.*, 7(1):27–43, 2011.

[7] A. Bar-Natan, L. Najt, and Z. Schutzman. The gerrymandering jumble: map projections permute districts' compactness scores. *Cartography and Geographic Information Science*, 47(4):321–335, 2020.

[8] C. Bazgan, B. Escoffier, and V. T. Paschos. Poly-APX- and PTAS-Completeness in standard and differential approximation. In R. Fleischer and G. Trippen, editors, *Algorithms and Computation*, pages 124–136, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[9] A. Becker and J. Solomon. Redistricting algorithms. In M. Duchin and O. Walch, editors, *Political Geometry: Rethinking Redistricting in the US with Math, Law, and Everything In Between*, pages 303–340. Springer International Publishing, Cham, 2022.

[10] A. Bernstein, J. Blikstad, T. Saranurak, and T.-W. Tu. Maximum flow by augmenting paths in $n^{2+o(1)}$ time. In *65nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2056–2077, 2024.

[11] S. Bhattacharya, P. Kiss, and T. Saranurak. Dynamic algorithms for packing-covering LPs via multiplicative weight updates. In N. Bansal and V. Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 1–47. SIAM, 2023.

[12] S. Bhore, S. Chakraborty, S. Jana, J. S. B. Mitchell, S. Pandit, and S. Roy. The balanced connected subgraph problem. *Discrete Appl. Math.*, 319:111–120, Oct. 2022.

[13] S. Bhore, S. Jana, S. Pandit, and S. Roy. The balanced connected subgraph problem for geometric intersection graphs. *Theoretical Computer Science*, 929:69–80, 2022.

[14] M. Böhm, A. Fazzone, S. Leonardi, and C. Schwiegelshohn. Fair clustering with multiple colors. arXiv 2002.07892, Feb. 2020.

[15] V. Bucarey, F. Ordóñez, and E. Bassaletti. Shape and balance in police districting. In *Applications of location analysis*, pages 329–347. Springer, 2015.

[16] L. Bursztyn, D. Cantoni, P. Funk, F. Schönenberger, and N. Yuchtman. Identifying the effect of election closeness on voter turnout: Evidence from Swiss referenda. *J. Eur. Econ. Assoc.*, 22(2):876–914, June 2023.

[17] T. M. Chan and S. Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discret. Comput. Geom.*, 48(2):373–392, 2012.

[18] Y. H. Chan and L. C. Lau. On linear and semidefinite programming relaxations for hypergraph matching. *Math. Program.*, 135(1-2):123–148, 2012.

[19] L. Chen, R. Kyng, Y. P. Liu, R. Peng, M. P. Gutenberg, and S. Sachdeva. Almost-linear-time algorithms for maximum flow and minimum-cost flow. *Commun. ACM*, 66(12):85–92, 2023.

[20] Y. Chen, Z.-Z. Chen, G. Lin, Y. Xu, and A. Zhang. Approximation algorithms for maximally balanced connected graph partition. *Algorithmica*, 83(12):3715–3740, 2021.

[21] A. Chhabra, K. Masalkovaitė, and P. Mohapatra. An overview of fairness in clustering. *IEEE Access*, 9:130698–130720, 2021.

[22] F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii. Fair clustering through fairlets. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 5036–5044, 2017.

[23] M. Chlebík and J. Chlebíková. Approximation hardness for small occurrence instances of NP-Hard problems. In *Algorithms and Complexity*, pages 152–164. Springer Berlin Heidelberg, 2003.

[24] J. Chlebíková. Approximating the maximally balanced connected partition problem in graphs. *Information Processing Letters*, 60(5):225–230, 1996.

[25] G. Chuang, O. Hanguir, and C. Stein. Drawing Competitive Districts in Redistricting. In G. N. Rothblum, editor, *5th Symposium on Foundations of Responsible Computing (FORC 2024)*, volume 295 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:22, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[26] J. Chuzhoy and S. Khanna. Maximum bipartite matching in $n^{2+o(1)}$ time via a combinatorial algorithm. In B. Mohar, I. Shinkar, and R. O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 83–94. ACM, 2024.

[27] C. Cirincione, T. A. Darling, and T. G. O'Rourke. Assessing South Carolina's 1990s congressional districting. *Political Geography*, 19(2):189–211, 2000.

[28] S. Clarke and J. Surkis. An operations research approach to racial desegregation of school systems. *Socio-Economic Planning Sciences*, 1(3):259–272, 1968.

[29] J. Clelland, H. Colgate, D. DeFord, B. Malmskog, and F. Sancier-Barbosa. Colorado in context: Congressional redistricting and competing fairness criteria in Colorado. *Journal of Computational Social Science*, 5(1):189–226, 2022.

[30] V. Cohen-Addad, P. N. Klein, and N. E. Young. Balanced power diagrams for redistricting. *arXiv preprint arXiv:1710.03358*, 2017.

[31] V. Cohen-Addad, P. N. Klein, and N. E. Young. Balanced centroidal power diagrams for redistricting. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '18, page 389–396, New York, NY, USA, 2018. Association for Computing Machinery.

[32] G. Cornuéjols, D. Hartvigsen, and W. Pulleyblank. Packing subgraphs in a graph. *Operations Research Letters*, 1(4):139–143, 1982.

[33] J. De Silva, B. Gales, B. Kagy, and D. Offner. An analysis of a fair division protocol for drawing legislative districts. *arXiv preprint arXiv:1811.05705*, 2018.

[34] D. DeFord, M. Duchin, and J. Solomon. A computational approach to measuring vote elasticity and competitiveness. *Statistics and Public Policy*, 7(1):69–86, 2020.

[35] D. DeFord, M. Duchin, and J. Solomon. Recombination: A family of Markov chains for redistricting. *Harvard Data Science Review*, 3(1):3, 2021.

[36] R. Diestel. *Graph Theory: 5th edition*. Springer-Verlag, © Reinhard Diestel, Feb. 2017.

[37] G. Ding and W. Zang. Packing cycles in graphs. *Journal of Combinatorial Theory, Series B*, 86(2):381–407, 2002.

[38] R. Duan, S. Pettie, and H. Su. Scaling algorithms for weighted matching in general graphs. *ACM Trans. Algorithms*, 14(1):8:1–8:35, 2018.

[39] E. Falkenauer and A. Delchambre. A genetic algorithm for bin packing and line balancing. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 1186–1192 vol.2, 1992.

[40] A. D. Franklin and E. Koenigsberg. Computed school assignments in a large district. *Operations Research*, 21(2):413–426, 1973.

[41] R. G. Fryer Jr and R. Holden. Measuring the compactness of political districting plans. *The Journal of Law and Economics*, 54(3):493–535, 2011.

[42] Z. Füredi, J. Kahn, and P. D. Seymour. On the fractional matching polytope of a hypergraph. *Comb.*, 13(2):167–180, 1993.

[43] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for general graph-matching problems. *J. ACM*, 38(4):815–853, 1991.

[44] M. R. Gary and D. S. Johnson. Computers and intractability: A guide to the theory of NP-completeness, 1979.

[45] N. Gillani, D. Beeferman, C. Vega-Pourheydarian, C. Overney, P. V. Hentenryck, and D. Roy. Redrawing attendance boundaries to promote racial and ethnic diversity in elementary schools. *Educational Researcher*, 52(6):348–364, 2023.

[46] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, Jan. 1985.

[47] M. Halldórsson and J. Radhakrishnan. Greed is good: approximating independent sets in sparse and bounded-degree graphs. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of Computing*, STOC '94, pages 439–448, New York, NY, USA, May 1994. Association for Computing Machinery.

[48] J. G. Hebert. *The Realist's Guide to Redistricting: Avoiding the Legal Pitfalls.* American Bar Association, 2010.

[49] S. W. Hess, J. Weaver, H. Siegfeldt, J. Whelan, and P. Zitlau. Nonpartisan political redistricting by computer. *Operations Research*, 13(6):998–1006, 1965.

[50] M. Jacobs and O. Walch. A partial differential equations approach to defeating partisan gerrymandering. *arXiv preprint arXiv:1806.07725*, 2018.

[51] X. Jia, K. Sheth, and O. Svensson. Fair colorful k-center clustering. In *Integer Programming and Combinatorial Optimization*, pages 209–222. Springer International Publishing, 2020.

[52] H. Jin. *Spatial optimization methods and system for redistricting problems.* PhD thesis, University of South Carolina, 2017.

[53] J. Kalcsics, S. Nickel, and M. Schröder. Towards a unified territorial design approach—applications, algorithms and GIS integration. *Top*, 13:1–56, 2005.

[54] P. Keevash and B. Sudakov. Packing triangles in a graph and its complement. *Journal of Graph theory*, 47(3):203–216, 2004.

[55] M. J. Kim. *Optimization approaches to political redistricting problems.* PhD thesis, The Ohio State University, 2011.

[56] Z. Landau, O. Reid, and I. Yershov. A fair division solution to the problem of redistricting. *Social Choice and Welfare*, 32(3):479–492, 2009.

[57] R. Lipton and R. Tarjan. A separator theorem for planar graphs. *SIAM J. Appl. Math.*, 36(2):177–189, Apr. 1979.

[58] T. Martinod, V. Pollet, B. Darties, R. Giroudeau, and J.-C. König. Complexity and inapproximability results for balanced connected subgraph problem. *Theoretical Computer Science*, 886:69–83, 2021.

[59] W. Mulzer and G. Rote. Minimum-weight triangulation is NP-hard. *J. ACM*, 55(2):1–29, May 2008.

[60] E. Najt, D. DeFord, and J. Solomon. Complexity and geometry of sampling connected graph partitions. *arXiv preprint arXiv:1908.08881*, 2019.

[61] G. Orfield and D. Jarvie. Black segregation matters: School resegregation and black educational opportunity. *Civil Rights Project-Proyecto Derechos Civiles*, 2020.

[62] W. Pegden, A. D. Procaccia, and D. Yu. A partisan districting protocol with provably nonpartisan outcomes. *arXiv preprint arXiv:1710.08781*, 2017.

[63] S. A. Plotkin, D. B. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 495–504. IEEE Computer Society, 1991.

[64] A. Samorodnitsky and L. Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, STOC '00, page 191–199, New York, NY, USA, 2000. Association for Computing Machinery.

[65] T. C. Schelling. Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186, 1971.

[66] J. van den Brand, Y. T. Lee, D. Nanongkai, R. Peng, T. Saranurak, A. Sidford, Z. Song, and D. Wang. Bipartite matching in nearly-linear time on moderately dense graphs. In S. Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 919–930. IEEE, 2020.

[67] J. B. Weaver and S. W. Hess. A procedure for nonpartisan districting: Development of computer techiques. *Yale Law Journal*, 73(2):288–308, 1963.

[68] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In J. M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 681–690. ACM, 2006.

A Omitted Proofs from Section 3

Throughout, we refer type-1 vertices as vertices that have non-zero weight p_1 and zero weight of p_2 and type-2 vertices as vertices with non-zero weight p_2 and zero weight of p_1.

Theorem 1. *The c-balanced districting problem is* NP*-hard, for both the case when the the districts are connected subgraphs and when the districts are required to be stars.*

Proof. We describe a polynomial reduction from the NP-hard problem of *exact set cover*: given a set S of n elements $x_1, x_2, \cdots, x_n$, and a family $\mathcal{S}$ of subsets $S_1, S_2, \cdots, S_m$, where $S_i \subseteq S$, is it possible to find $\mathcal{S}' \subseteq \mathcal{S}$ such that every element in S is covered by exactly one set in $\mathcal{S}'$.

We construct a bipartite graph G where we place one vertex for each element x_i and one vertex for each set S_j. We connect edges between x_i with S_j if and only if $x_i \in S_j$. We assign $p_1(x_i) = 1$ and $p_2(x_i) = 0$, $\forall i$. And we assign $p_1(S_j) = 0$ and $p_2(S_j) = (c-1)|S_j|$, $\forall j$. The only c-balanced districts in this graph are stars, each rooted at a vertex/block S_j with all neighbors in S. Therefore if there is an exact set cover, then there is a solution to the c-balanced districting problem in G with total weight of cn. Otherwise, the best c-balanced districting problem in G has total weight no greater than $c(n-1)$. □

Theorem 2. *The c-balanced districting problem is* NP*-hard, when G is a planar graph with maximum degree* 3*, each district has rank-*3 *(i.e., with at most three blocks), and the districts must be stars.*

Proof. We use a reduction from *planar 1-in-3SAT* [59]. This is a problem of n Boolean variables $x_1, x_2, \ldots, x_n$ and m clauses each with three literals, and the incidence graph of how the variables appear in the clauses is planar. The problem answers YES if there is an assignment to the Boolean variables such that each clause is satisfied by exactly one literal.

We realize the planar 1-in-3SAT instance by a planar graph. In the graph we have two types of vertices, type-1 vertices which has non-zero weight p_1 and zero weight of p_2 and type-2 vertices with non-zero weight p_2 and zero weight of p_1. We take $c = 2$ thus each district must have an equal amount of the two types of weight.

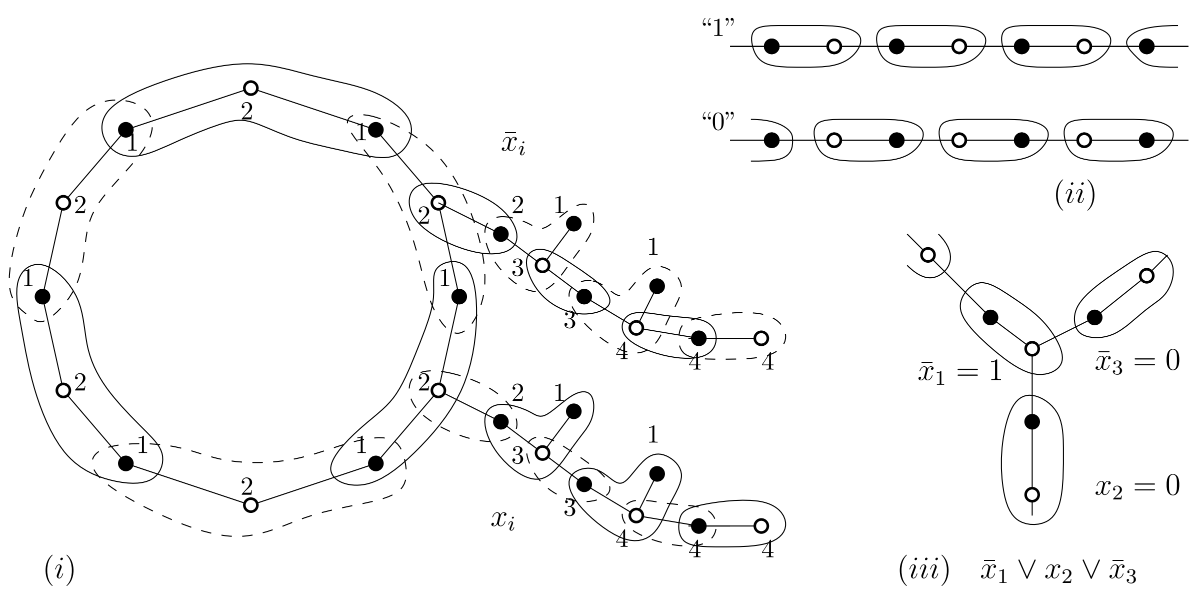

Figure 1: (i) A variable gadget for x_i. Solid vertices are type-2 vertices and hallow vertices are type-1 vertices. Here we use $m = 3$ in this example and one clause has x_i and one has $\bar{x}_i$. Thus there are two tendrils connect to clauses. If $x_i = 1$ we choose solid districts; otherwise we choose dashed districts. (ii) Implementation of an edge from a variable x_i to a clause C_j where x_i appears. If the chain is propagating "1", the districts on the chain starting from a variable gadget have a type-2 vertex followed by a type-1 vertex; otherwise the districts on the chain have a type-1 vertex followed by a type-2 vertex. (iii) The clause gadget $x_1 \cup x_2 \cup \bar{x}_3$. Exactly one of $x_1, x_2, \bar{x}_3$ is assigned 1 which means that the center type-2 vertex is covered by exactly one district.

Figure 1 (i) shows the gadget for a variable x_i. The center of this gadget is a cycle of $4m$ vertices, arranged on a cycle with alternating type-1 vertices (hallow) and type-2 vertices (solid).

The type-1 vertices have weight of $p_1 = 2$ and type-2 vertices have weight of $p_2 = 1$. From a type-1 vertex we may grow one tendril that leads to a clause gadget, if this vertex appears in the clause. There are a total of $2m$ type-1 vertices where the tendrils of m of them may lead to clauses where x_i appears in the original form; and the other m of them may lead to clauses where $\bar{x}_i$ appears. These are arranged in an alternating manner along the cycle. If x_i appears in only k clauses, there are only k tendrils. Those tendrils that connect to clause gadgets carry alternating type-1 and type-2 vertices. The weights are labeled nearby the vertices in the Figure.

The only balanced district we can find on the cycle is a star centered at a type-1 vertex v with two type-2 leaves that are neighbors of v on the cycle. Further, we could use two sets of districts (each set has m balanced districts) to cover all type-2 vertices on the cycle. In the figure these two sets of districts are shown in solid sets and dashed sets respectively. They appear in alternating order along the cycle. Using either set we cover all type-2 vertices on the cycle and leave out precisely m type-1 vertices uncovered. If we choose the solid set it means we assign 1 to this variable. If we choose the dashed set it means we assign 0 to this variable. Either way all type-2 vertices along the variable cycles are covered. How the type-1 vertices are covered (or not) depends on the assignment of the corresponding variable.

The choice of the districts (corresponding to the assigned Boolean value to x_i) is propagated to the clause set. The implementation of an edge connecting a variable x_i to a clause C_j is by a *path* of alternating type-1 vertices and type-2 vertices. See Figure 1 (ii). Each path starts from a type-1 vertex on the variable gadget and stops at a type-1 vertex at the center of a clause gadget. The path can be propagating a value of '1' if the covering tree takes a type-2 vertex followed by a type-1 vertex along the direction from the variable gadget to the clause gadget (solid before hallow), or '0' if swapped. When the first type-1 vertex along a tendril is covered by the variable assignment choice on the variable gadget (meaning the corresponding literal is taken as a 1), the districts on the tendril will be propagating a "1" to the clause.

One subtle issue of Figure 1 (i) is the covered weights of type-1 vertices, because the number of times a variable x_i appears in the clauses can be different from that of $\bar{x}_i$. Suppose we take $x_i = 1$ (i.e., the solid districts), the tendrils corresponding to $\bar{x}_i$ grab a type-1 (hallow) vertex on the cycle, while the type-1 vertices in the dashed districts on the variable gadget that are not attached to any tendrils will not be covered. Thus, the weight of type-1 vertices in the cycle and tendrils uncovered by $x_i = 1$ is $2(m - k(x_i))$, with $k(x_i)$ as the number of times that $\bar{x}_i$ appears in clauses. To make sure that we always leave out the same total weight uncovered regardless of $x_i = 1$ or $x_i = 0$, we amend the beginning part of the tendrils such that if it is propagating a "0", we leave out an extra weight of two not covered. We achieve this by gradually increasing the node weight from 2 on the cycle to 4 with two attached vertices of each of type-2 weight 1. Figure 1 (i) shows how the two extra vertices are attached. This way, regardless of whether $x_i = 1$ and $x_i = 0$, the total weights not covered by the variable gadget along with the beginning parts of the tendrils are exactly $2m$.

The clause gadget is implemented by a single type-1 vertex where three tendrils (corresponding to three literals) meet. If x_i (or $\bar{x}_i$) appears in the clause, we connect a path from a type-1 vertex covered by a solid (or dashed) district of the variable gadget x_i. Thus if the solid district is chosen, the path from x_i propagates an assignment of 1 to the clause and covers the center. If the dashed tree cover is chosen, the path from x_i propagates an assignment of 0 which cannot cover the center of the clause gadget. The center is covered by the (unique) path that is assigned "1". See Figure 1 (iii) for an example.

In summary, if the planar 1-in-3SAT is satisfiable, all m type-2 vertices of the clause gadgets are covered by disjoint balanced districts. If the planar 1-in-3SAT cannot be satisfied, any balanced districting will leave at least some type-2 vertices not covered. This shows that the c-balanced districting problem in a planar graph is NP-hard. □

Theorem 3. *The c-balanced districting problem is* NP*-hard for any $c \geq 2$, when G is a complete graph or a tree. This holds for both the case when the the districts are connected subgraphs and when the districts are required to be stars.*

Proof. We start from an instance of SubsetSum and turn it into an instance of c-balanced districting problem. In SubsetSum, there are n numbers $x_1, x_2, \cdots, x_n$ and the goal is to ask if any subset sums up to $X = \sum_i x_i/2$. Now we create a complete graph G with $n+1$ vertices including a vertex v_0 of type-2 weight of X, and type-1 weight of 0, and n vertices $v_1, \cdots v_n$ with v_i holding type-1 weight of $(c-1) \cdot x_i$ and type-2 weight of 0.

Therefore, if the SubsetSum answers positively, there is a star with root v_0 and the vertices in the SubsetSum solution that achieves the maximum possible covered population of $c \cdot X$. On the other hand, if c-balanced districting problem returns a solution with covered population of $c \cdot X$, then the answer to the SubsetSum instance is true.

Similarly, we can reduce a SubsetSum instance to a star, where v_0 is the root of the tree, and $v_1, \cdots, v_n$ are leaves. The same argument follows. □

Theorem 4. *The c-balanced districting problem does not have an $n^{1/2-\delta}$-approximation (for any constant $\delta > 0$) in a general graph unless* P = NP*. On a graph with maximum degree Δ, one cannot approximate the c-balanced districting problem within a factor of $\Delta/2^{O(\sqrt{\Delta})}$ assuming* P ≠ NP*, and $O(\Delta/\log^2 \Delta)$ assuming the Unique Games Conjecture (UGC). Even if Δ is a constant, the problem is* APX*-hard. These statements hold when the districts must be stars and when the centers of the stars are limited to a subset of vertices.*

Proof. We take an instance of maximum independent set problem and turn it into an instance of c-balanced districting problem. Given a graph $G = (V, E)$, $n = |V|$ and $m = |E|$. Denote by Δ the maximum degree of G. For each vertex $v \in V$, we create a vertex v' in graph G' with type-1 weight of $(c-1)\Delta$ and type-2 weight of 0. There are $n = |V| = |V'|$ such "type-1 vertices". We also have a set of "type-2 vertices" V'' with type-2 weight of 1 and type-1 weight of 0. Each vertex $v' \in V'$ has exactly Δ type-2 neighbors. If u, v has an edge in G, in G', u' and v' share one type-2 vertex, which corresponds to the edge between u, v in G. See Figure 2. If the degree of u is less than Δ, the corresponding vertex in G' may have some dangling (degree-1) type-2 vertices. The total number of type-2 vertices is $n\Delta - m$. Thus the total number of vertices in G' is $n(\Delta+1) - m$ and the number of edges in G' is $n\Delta$. In order for a type-1 vertex u' to be covered, all its type-2 neighbors must be used. Thus a maximum independent set S in G means we can cover all corresponding vertices of S in G' as well as all their type-2 neighbors, leading to a total coverage population of $|S|\Delta$. Similarly, if we can find a c-balanced districting problem in G', the type-1 vertices that are covered in c-balanced districts cannot share any common type-2 neighbors, and therefore the corresponding vertices in G must be independent. This reduction works when the district must be a star.

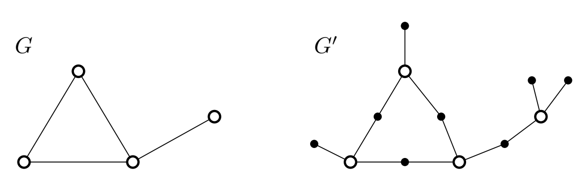

Figure 2: Graph G and G'. $\Delta = 3$ in this instance. The hallow vertices of G' have type-1 weight of $3(c-1)$ and the solid vertices of G' have type-2 weight of 1.

This reduction shows hardness of approximation, as an α-approximation for maximum independent set means an α-approximation for c-balanced districting problem, for any c. The maximum independent set cannot be approximated by a factor of $n^{1-\delta}$ for any constant $\delta > 0$ on general graph [8, 68]. If we have an approximation algorithm for the districting problem with approximation factor $O(N^{1/2-\delta})$ with N as the number of vertices in the districting graph G', by the reduction $N = O(n\Delta)$ and this gives an $O((n\Delta)^{1/2-\delta}) = O(n^{1-2\delta})$ for the maximum independent set problem on G, since $\Delta < n$, which is impossible unless $\mathsf{P} = \mathsf{NP}$.

As the maximum degree in both G and G' is Δ, approximating the balanced districting problem in G' with some factor depending on Δ gives the same approximation factor for the maximum independent set in G. For bounded degree graphs, the maximum independent set has a constant approximation [47], but is APX-complete [23] and cannot expect an approximation ratio better than $\Delta/2^{O(\sqrt{\Delta})}$ unless $\mathsf{P} = \mathsf{NP}$ [64]. Further, assuming the Unique Games Conjecture, one cannot approximate the maximum independent set problem within a factor of $O(\Delta/\log^2 \Delta)$ [6]. These (conditional) hardness results extend to balanced star districting problem on graphs with maximum degree Δ. This finishes the argument. □

B A Whack-A-Mole Algorithm For c-Balanced Star Districting

B.1 The MWU Framework

We first recap the classical multiplicative weight update framework.

Consider the following online process. There are $n \in \mathbb{N}$ experts and there is a weight vector $y^{(1)} = \frac{1}{n} \cdot \mathbf{1}^n$. Let $\rho \in \mathbb{R}_{>0}$ be a positive value that represents the bound of an expert's gain in a single round. Let $T := \lceil \rho\varepsilon^{-2} \ln n \rceil$ be the target number of rounds. For each round $t = 1, 2, \ldots, T$, the process receives a *reward* vector $g^{(t)} = [g_1^{(t)}, \ldots, g_n^{(t)}] \in [-\rho, \rho]^n$. Then, the process computes a temporary vector $\hat{y}_i^{(t+1)} := (1 + \varepsilon \cdot g_i^{(t)}/\rho) \cdot y_i^{(t)}$ for all $i = 1, 2, \ldots, n$, and then normalizes it, obtaining $y^{(t+1)} := \hat{y}^{(t+1)}/\|\hat{y}^{(t+1)}\|$. After the execution of all T rounds, we have the following guarantee (from [5, Corollary 2.6]).

Theorem 13. *Fix any i. Let $G_i := \sum_{t=1}^{T} g_i^{(t)}$ be the sum of all rewards of the i-th expert throughout the process. Let $G_{\mathrm{ALG}} := \sum_{t=1}^{T} \langle y^{(t)}, g_i^{(t)} \rangle$ be the expected total gain, if in each round t the prediction algorithm randomly picks an expert according to the distribution $y^{(t)}$, and follows that expert's decision. Then,*

$$\frac{G_i}{T} \le (1 + \varepsilon)\frac{G_{\mathrm{ALG}}}{T} + \varepsilon.$$

B.2 The Whack-A-Mole Algorithm

Let us focus on the dual linear program. Without loss of generality we assume that each c-balanced district S has positive population, namely, $w(S) \ge 1$. Let μ be the binary searched value to the optimal objective value of our linear programs. It is straightforward to check that $\mu \in [1, w(G)]$, where $w(G)$ is the total weight of all vertices in the graph. Given that the re-weighted linear program where each coefficient of the form $1/w(S)$ is replaced with $\mu/w(S)$, the goal of the whack-a-mole algorithm now is to either find a feasible primal solution of value $\ge 1 - 2\varepsilon$, or a feasible dual solution of value $\le 1 + \varepsilon$. This ensures an $(1 + 3\varepsilon)$-approximation in the end. If we aim for an $(1 + \varepsilon')$-approximation with a specific $\varepsilon' > 0$, then we can always initialize $\varepsilon := \varepsilon'/3$.

We first describe the slow version of the whack-a-mole algorithm. This algorithm simulates the prediction game round by round with a total of $T := \mu\varepsilon^{-2} \ln n$ rounds. We set $\rho = \mu$. The algorithm initializes $x_S = 0$ for all $S \in \mathcal{S}$ and $y_v = 1/n$ for all $v \in V$. In each round t, the whack-a-mole algorithm invokes the separation oracle and finds a weakly violating c-balanced district S with

$$\frac{\mu}{w(S)} \cdot \langle y^{(t)}, \mathbf{1}_S \rangle < 1 - \frac{\varepsilon}{2}. \tag{11}$$

The whack-a-mole algorithm then sets a reward vector $g^{(t)} := \frac{\mu}{w(S)} \mathbf{1}_S$. Thus, $\langle y^{(t)}, g^{(t)} \rangle < 1 - \varepsilon/2$. Meanwhile, the algorithm also increases the variable x_S by 1. This process repeats for round $t = 1, 2, \ldots,$ until either (1) $t = T$, or (2) there is no strongly violating c-balanced district anymore.

In case (1), the algorithm returns $x'_S \leftarrow (1 - 2\varepsilon)x_S/T$ as a feasible primal solution. Note that now $\sum_S x'_S = 1 - 2\varepsilon$. This indicates that $OPT \geq \mu$. In case (2), the algorithm returns a feasible dual solution $y'_v \leftarrow (1 - \varepsilon)^{-1} \cdot y_v^{(t)}$, where t refers to the last round of the WMU step. Note that now $\sum_v y'_v = (1 - \varepsilon)^{-1}$. This indicates that $OPT \leq (1 + 2\varepsilon)\mu$ whenever $\varepsilon \leq 1/2$. Now, we show that the algorithm does halt with a feasible (either primal or dual) solution.

Lemma B.1. *If the algorithm ends in case (1), then $\{x'_S\}$ is a feasible primal solution.*

Proof. In case (1), T rounds of the MWU steps are performed. Hence, we are able to utilize Theorem 13 to validate the primal constraints. For every $v \in V$, the constraint in (Primal) states that

$$\begin{aligned}
\sum_{S \in \mathcal{S}: S \ni v} \frac{\mu}{w(S)} x'_S &= (1 - 2\varepsilon) \cdot \frac{1}{T} \sum_{S \in \mathcal{S}: S \ni v} \frac{\mu}{w(S)} x_S \\
&= (1 - 2\varepsilon) \cdot \frac{1}{T} \sum_{S \in \mathcal{S}: S \ni v} \frac{\mu}{w(S)} \cdot (\text{\# of rounds where } S \text{ is chosen}) \\
&= (1 - 2\varepsilon) \cdot \frac{1}{T} \sum_{t=1}^{T} \frac{\mu}{w(S^{(t)})} \mathbb{I}[g_v^{(t)} \neq 0] && (\mathbb{I}[\cdot] \text{ is the indicator function}) \\
&= (1 - 2\varepsilon) \cdot \frac{1}{T} \sum_{t=1}^{T} \frac{\mu}{w(S^{(t)})} \cdot g_v^{(t)} \cdot \frac{w(S^{(t)})}{\mu} && (\text{by definition of } g_v^{(t)}) \\
&= (1 - 2\varepsilon) \cdot \frac{1}{T} \sum_{t=1}^{T} g_v^{(t)} = (1 - 2\varepsilon) \frac{G_v}{T} \\
&\leq (1 - 2\varepsilon) \cdot \left(\varepsilon + (1 + \varepsilon) \frac{G_{\text{ALG}}}{T} \right) && (\text{By Theorem 13}) \\
&= (1 - 2\varepsilon) \cdot \left(\varepsilon + \frac{1 + \varepsilon}{T} \sum_{t=1}^{T} \langle g^{(t)}, y^{(t)} \rangle \right) \\
&\leq (1 - 2\varepsilon) \cdot \left(\varepsilon + \frac{1 + \varepsilon}{T} \sum_{t=1}^{T} (1 - \frac{\varepsilon}{2}) \right) && (\text{By Equation (11)}) \\
&\leq (1 - 2\varepsilon) \cdot (1 + \frac{3}{2}\varepsilon) \leq 1
\end{aligned}$$

□

Lemma B.2. *If the algorithm ends in case (2), then $\{y'_v\}$ is a feasible dual solution.*

Proof. In case (2), the separation oracle certifies that there is no strongly violating constraints. Thus, from the definition of a strongly violation constraint (Equation (3)), we know that for all c-balanced star district S, we have $\sum_{v \in S} y_v^{(t)} \geq (1 - \varepsilon) \cdot w(S)$, which implies that $\sum_{v \in S} y'_v \geq w(S)$. Thus, $\{y'_v\}$ is a feasible dual solution. □

B.3 Speeding Up The Algorithm

The previous algorithm requires T rounds of WMU steps. Unfortunately, $T = \lceil \mu\varepsilon^{-2}\ln n\rceil$ and μ can be as large as $w(G)$. A straightforward implementation requires $\Omega(w(G))$ time, even if we are provided with a polynomial time separation oracle. From the description of our algorithm, we observe that without normalization, the sum $\sum_v y_v$ is non-decreasing. The main idea to speed up the MWU algorithm (provided in [11]), is to perform a lazy normalization of the dual variables: the algorithm normalizes $\{y_v\}$ whenever its sum exceeds $1+\varepsilon$. The MWU rounds between two normalizations is then called a *phase*. Within the same phase, the algorithm is allowed to perform multiple MWU steps on the same violating constraint S *at once* until one of the following three situations happen: (1) this constraint becomes satisfied, (2) the total number of steps now reaches T, or (3) the sum of dual variables exceeds $1+\varepsilon$ so the algorithm enters a new phase. This subroutine is called ENFORCE in [11]. We are now able to give a polynomial upper bound to the number of phases in total.

Lemma B.3. *The number of phases is at most $O(\varepsilon^{-2}\log n)$.*

Proof. There are $T := \lceil \mu\varepsilon^{-2}\ln n\rceil$ MWU rounds. Let $\hat{y}_v^{(t)}$ be the unnormalized dual variables and let $\|\hat{y}^{(t)}\| = \sum_v \hat{y}_v^{(t)}$ be the sum at the beginning of the round t. Initially, the sum of all dual variables is $\sum_v \hat{y}_v^{(1)} = 1$. Within each round t, a (weaker) violating constraint S is chosen such that

$$\sum_{v\in S} \hat{y}_v^{(t)} \le (1-\varepsilon/2)\cdot \frac{w(S)}{\mu}\cdot \|\hat{y}^{(t)}\|.$$

A MWU step multiplies each $\hat{y}_v^{(t)}$ by $(1+\varepsilon\cdot g_v^{(t)}/\mu)$ for $v\in S$. Notice that $g_v^{(t)} := \mu/w(S)$ for all $v\in S$. This implies that the absolute increase of the unnormalized dual variables can be bounded by

$$\begin{aligned}
\|\hat{y}^{(t+1)}\| - \|\hat{y}^{(t)}\| &\le \sum_{v\in S}\frac{\varepsilon}{\mu}\cdot g_v^{(t)}\cdot \hat{y}_v^{(t)}\\
&= \frac{\varepsilon}{\mu}\cdot\frac{\mu}{w(S)}\cdot\sum_{v\in S}\hat{y}_v^{(t)}\\
&\le \frac{\varepsilon}{\mu}\cdot\frac{\mu}{w(S)}\cdot(1-\varepsilon/2)\cdot\frac{w(S)}{\mu}\cdot\|\hat{y}_v^{(t)}\|\\
&\le \frac{\varepsilon}{\mu}\|\hat{y}_v^{(t)}\|.
\end{aligned}$$

Thus, each MWU step increases the total sum by a factor of at most $1+\varepsilon/\mu$. Therefore,

$$\begin{aligned}
\|\hat{y}^{(T)}\| &\le (1+\varepsilon/\mu)^T = (1+\varepsilon/\mu)^{\lceil \mu\varepsilon^{-2}\ln n\rceil}\\
&\le e^{(\ln n)/\varepsilon+1} = e\cdot n^{1/\varepsilon}.
\end{aligned}$$

Notice that the algorithm is triggered to start a new phase whenever the current normalized sum of the dual variables $\sum_v y_v \ge 1+\varepsilon$. Hence, the total number of phases can be bounded by

$$\log_{1+\varepsilon}\left(e\cdot n^{1/\varepsilon}\right) = O(\varepsilon^{-2}\ln n),$$

as desired. □

B.4 Separation Oracle: Proof of Lemma 4.1

Lemma 4.1. *Given an input instance $G = (V, E, (p_1, p_2))$, re-weighted values $w' : V \to \{0\} \cup [\frac{1}{w(G)}, w(G)]$, and dual variables $y'_v \in [n^{-(1+1/\varepsilon)}, 1+\varepsilon]$ for each vertex $v \in V$, there exists an algorithm that either reports that $\mathcal{S}_{\text{violate}} = \emptyset$, or returns a c-balanced district S such that $\sum_{v\in S} y'_v < (1-\varepsilon/2)w'(S)$ and $w'(S) \geq \frac{1}{2}w'(S_{\max})$, where $S_{\max} = \arg\max_{S\in\mathcal{S}_{\text{violate}}} w'(S)$ is a violating c-balanced district with the maximum value. The algorithm runs in $O(\varepsilon^{-6}n^6(\log n)(\log w(G))^4)$ time.*

Proof. We generalize Algorithm 1 to accommodate dual variables. In particular, we will maintain a candidate list L with the following property: For any c-balanced district S, there exists a district $T \in L$ such that (1) $(p_1(S), p_2(S))$ is an $(\varepsilon/10)$-approximate of $(p_1(T), p_2(T))$, and (2) $\sum_{v\in S} w'(v)/\sum_{v\in T} w'(v) \in [e^{-\varepsilon/10}, e^{\varepsilon/10}]$. Recall that from the proof of Theorem 7 we defined that a pair of numbers (q_1, q_2) is an ε-approximate to (q'_1, q'_2) if both $q_1/q'_1, q_2/q'_2 \in [e^{-\varepsilon}, e^{\varepsilon}]$. The above property suggests that we add a third dimension for y'_v to the list, and modify the trimming algorithm slightly — we will not trim the solution if their $\sum_{v\in S} y'_v$ values are too far from each other.

We now prove that the final list contains a c-balanced district that is a weakly violating constraint. Consider the population of commodities $(p_1(S_{\max}), p_2(S_{\max}))$ of $S_{\max}$. Without loss of generality, assume that $p_1(S_{\max}) \geq p_2(S_{\max})$. Then, by the property we stated above, there exists a district $S \in L$ that satisfies:

$$\begin{aligned}
(c-1)p_2(S) - p_1(S) &\geq (c-1)p_2(S_{\max}) - p_1(S_{\max}) && \text{(maintained by Algorithm 1)}\\
&\geq 0, \text{ and}\\
(c-1)p_1(S) - p_2(S) &\geq (1-\varepsilon/10)(c-1)p_1(S_{\max}) - (1+\varepsilon/10)p_2(S_{\max})\\
&\geq ((1-\varepsilon/10)(c-2) - (\varepsilon/5))\, p_2(S_{\max})\\
&\geq 0 && \text{(whenever } \varepsilon \leq \tfrac{c-2}{c})
\end{aligned}$$

The above inequality shows that S is indeed c-balanced. Furthermore, we have $\sum_{v\in S} y'_v/\sum_{v\in S_{\max}} y'_v \in [e^{-\varepsilon/10}, e^{\varepsilon/10}]$. Thus, we are able to show that S is a weakly violating constraint:

$$\begin{aligned}
\sum_{v\in S} y'_v &\leq e^{\varepsilon/10} \cdot \sum_{v\in S_{\max}} y'_v\\
&\leq e^{\varepsilon/10} \cdot (1-\varepsilon) \cdot w'(S_{\max}) && (S_{\max} \text{ is strongly violating})\\
&\leq e^{\varepsilon/10} \cdot (1-\varepsilon) \cdot e^{\varepsilon/10} \cdot w'(S)\\
&\leq e^{\varepsilon/5} \cdot (1-\varepsilon) \cdot w'(S) \leq (1-\varepsilon/2) \cdot w'(S). && (\varepsilon > 0)
\end{aligned}$$

On the other hand, we have

$$w'(S) \geq e^{-\varepsilon/10} \cdot w'(S_{\max}) \geq \frac{1}{2} w'(S_{\max}),$$

certifying that the output S satisfies all the constraints from the lemma statement.

Let us now analyze the runtime of the algorithm. It suffices to analyze the number of scales at the new dimension. Since each value y'_v is at least $n^{-(1+1/\varepsilon)}$ and is at most $1+\varepsilon$, the number of scales in the third dimension can be bounded by

$$\log_{e^{\varepsilon/10}} \frac{1+\varepsilon}{n^{-(1+1/\varepsilon)}} = \frac{\ln(1+\varepsilon) + (1+1/\varepsilon)\ln n}{\varepsilon/10} = O(\varepsilon^{-2}\ln n).$$

Together with the analysis in Theorem 7, the runtime of this generalized algorithm for the complete graph, including maintaining a solution, is $O(\varepsilon^{-6}n^6(\log n)(\log w(G))^4)$. □

B.5 Piecing Everything Together: Proof of Theorem 6

Correctness. The correctness comes from the whack-a-mole framework [11], the implementation to the separation oracle (Lemma 4.1), and the potential method analyzed in Section 4.2.

Runtime Analysis. We have $O(\varepsilon^{-1}\log w(G))$ for binary search up to an $(1+\varepsilon)$-approximation. For each guess μ, we run the whack-a-mole algorithm which has $O(\varepsilon^{-2}\log n)$ phases and within each phase there are at most $O(n^2/\varepsilon)$ calls to the separation oracle by Equation (7). Each call to the separation oracle takes $O(\varepsilon^{-6}n^6(\log n)(\log w(G))^4))$ by Lemma 4.1. Therefore, the entire algorithm runs in time

$$O(\varepsilon^{-10}n^8(\log n)^2(\log w(G))^5) = \mathrm{poly}(n, 1/\varepsilon, \log w(G)).$$

Number of Non-Zero Primal Variables. It is probably worth to note that the number of non-zero terms does not involve any $\log w(G)$ terms. In particular, the algorithm uses the last returned feasible primal solution as the approximated solution. The number of non-zero terms can then be bounded by the number of phases multiplied by the number of separation oracle calls per phase, which is at most $O(\varepsilon^{-3}n^2\log n)$.

B.6 Proof of Randomized Rounding

Lemma B.4. *The randomized rounding algorithm from the fractional LP solution produces an expected weight for output districting I as*

$$\mathbf{E}[w(I)] \geq \sum_{S\in\mathcal{S}_{\mathrm{LP}}} w(S)\frac{x_S}{\tau} - \sum_{A,B\in\mathcal{S}_{\mathrm{LP}}:\ A\cap B\neq\emptyset} \min(w(A), w(B))\frac{x_Ax_B}{\tau^2}\ .$$

Proof. Consider a district S with non-zero value x_S, it is selected into I only if two events happen: (1) the coin flip with probability x_S/τ turns out to be true; and (2) all the districts with value at least x_S are not included in I – their coin flips are false. The probability of both events happening is

$$\frac{x_S}{\tau}\cdot\prod_{A\in\mathcal{S}_{\mathrm{LP}}:A\cap S\neq\emptyset, w(A)\geq w(S)}\left(1-\frac{x_A}{\tau}\right) \geq \frac{x_S}{\tau}\cdot\left(1-\sum_{A\in\mathcal{S}_{\mathrm{LP}}:A\cap S\neq\emptyset, w(A)\geq w(S)}\frac{x_A}{\tau}\right)$$

Now, by linearity of expectation, we have

$$\begin{aligned}\mathbf{E}[w(I)] &\geq \sum_{S\in\mathcal{S}_{\mathrm{LP}}} w(S)\frac{x_S}{\tau}\cdot\left(1-\sum_{A\in\mathcal{S}_{\mathrm{LP}}:A\cap S\neq\emptyset, w(A)\geq w(S)}\frac{x_A}{\tau}\right)\\ &= \sum_{S\in\mathcal{S}_{\mathrm{LP}}} w(S)\cdot\frac{x_S}{\tau} - \sum_{S,A\in\mathcal{S}_{\mathrm{LP}}:A\cap S\neq\emptyset, w(A)\geq w(S)} w(S)\cdot\frac{x_Sx_A}{\tau^2}\\ &= \sum_{S\in\mathcal{S}_{\mathrm{LP}}} w(S)\cdot\frac{x_S}{\tau} - \sum_{A,B\in\mathcal{S}_{\mathrm{LP}}:\ A\cap B\neq\emptyset}\min(w(A), w(B))\cdot\frac{x_Ax_B}{\tau^2}\ .\end{aligned}$$

□

For any $\delta \geq 0$, let $\mathcal{S}_{\geq\delta}$ be the set of all districts $S \subseteq \mathcal{S}_{\mathrm{LP}}$ whose weight is at least δ. The following lemma connects the unweighted correlation between overlapped districts and the expected approximation ratio to the randomized rounding algorithm.

Lemma B.5. *Let $\tau \in \mathbb{R}_{>0}$ be a fixed value. Suppose that for all $\delta > 0$,*

$$\sum_{A,B\in\mathcal{S}_{\geq\delta}:A\cap B\neq\emptyset} x_A x_B \leq (\tau/2)\cdot \sum_{S\in\mathcal{S}_{\geq\delta}} x_S, \tag{12}$$

then $\mathbf{E}[w(I)] \geq \frac{1}{2\tau}\sum_S w(S)\cdot x_S$.

Proof. We first sort all districts in $\mathcal{S}_{\text{LP}}$ in the non-increasing order of weights. Let $S_1, S_2, \ldots, S_t$ be such a list. For each district S_i, its weight $w(S_i)$ can be written as

$$w(S_i) = \sum_{j=i}^{t}(w(S_j) - w(S_{j+1})).$$

Here for convenience we define $w(S_{t+1}) = 0$. Using the above expression, we are able to establish that

$$\begin{aligned}
&\sum_{A,B\in\mathcal{S}_{\text{LP}}:\ A\cap B\neq\emptyset} \min(w(A),w(B))\cdot\frac{x_A x_B}{\tau^2} \\
&= \sum_{i=1}^{t}\sum_{\ell=1}^{i-1}\mathbb{I}[S_\ell\cap S_i\neq\emptyset]\cdot w(S_i)\cdot\frac{x_{S_i}x_{S_\ell}}{\tau^2} \\
&= \sum_{i=1}^{t}\sum_{\ell=1}^{i-1}\mathbb{I}[S_\ell\cap S_i\neq\emptyset]\cdot\left(\sum_{j=i}^{t}(w(S_j)-w(S_{j+1}))\right)\cdot\frac{x_{S_i}x_{S_\ell}}{\tau^2} \\
&= \sum_{j=1}^{t}\Big(w(S_j)-w(S_{j+1})\Big)\cdot\left(\sum_{i=1}^{j}\sum_{\ell=1}^{i-1}\mathbb{I}[S_\ell\cap S_i\neq\emptyset]\cdot\frac{x_{S_i}x_{S_\ell}}{\tau^2}\right) \\
&= \sum_{j=1}^{t}\Big(w(S_j)-w(S_{j+1})\Big)\cdot\left(\frac{1}{\tau^2}\cdot\sum_{A,B\in\mathcal{S}_{\geq w(S_j)}:\ A\cap B\neq\emptyset} x_A x_B\right) \\
&\leq \sum_{j=1}^{t}\Big(w(S_j)-w(S_{j+1})\Big)\cdot\left(\frac{1}{\tau^2}\cdot\frac{\tau}{2}\cdot\sum_{S\in\mathcal{S}_{\geq w(S_j)}} x_S\right) && \text{(by (12))} \\
&= \frac{1}{2\tau}\sum_{j=1}^{t}\sum_{i=1}^{j}\Big(w(S_j)-w(S_{j+1})\Big)\cdot x_{S_i} \\
&= \frac{1}{2\tau}\sum_{i=1}^{t} x_{S_i}\cdot\sum_{j=i}^{t}\Big(w(S_j)-w(S_{j+1})\Big) \\
&= \frac{1}{2\tau}\sum_{i=1}^{t} w(S_i)\cdot x_{S_i}
\end{aligned}$$

Finally, we have

$$\begin{aligned}
\mathbf{E}[w(I)] &\geq \sum_{S\in\mathcal{S}_{\text{LP}}} w(S)\frac{x_S}{\tau} - \sum_{A,B\in\mathcal{S}_{\text{LP}}:\ A\cap B\neq\emptyset}\min(w(A),w(B))\frac{x_A x_B}{\tau^2} && \text{(by Lemma B.4)} \\
&\geq \frac{1}{\tau}\sum_{S\in\mathcal{S}_{\text{LP}}} w(S)x_S - \frac{1}{2\tau}\sum_{S\in\mathcal{S}_{\text{LP}}} w(S)x_S
\end{aligned}$$

$$= \frac{1}{2\tau} \sum_{S \in \mathcal{S}_{\mathrm{LP}}} w(S) x_S$$

as desired. □

B.7 Outerplanar Graphs

In this section we devote in proving the following lemma.

Lemma 4.10. *Let G be an outerplanar graph. Suppose that $\{x_S\}$ are primal variables obtained by Theorem 6. Then, $\sum_{A \cap B \neq \emptyset} x_A x_B \leq O(1) \cdot \sum x_S$.*

Let G be a connected outerplanar graph. We begin with the classical result [36].

Fact B.6. *A graph G is outerplanar if and only if G does not have $K_{2,3}$ and K_4 as minor.*

Let $r \in V$ be any vertex in G. By considering shortest distances of vertices on G from r, we are able to partition V into layers $V = L_0 \cup L_1 \cup \cdots \cup L_d$, where $L_i = \{v \in V \mid \mathrm{dist}_G(v, r) = i\}$. Notice that algorithmically we can construct the sets $\{L_i\}$ using a single BFS, but here we do not need them. But with the BFS tree in mind, we obtain the following fact.

Fact B.7. *Consider any three vertices from the same layer $x, y, z \in L_i$. Then $\{r, x, y, z\}$ form a claw minor ($K_{1,3}$-minor) in G.*

The following lemma is crucial for proving Lemma 4.10.

Lemma B.8. *For each layer L_i, $i \geq 1$, the induced subgraph $G[L_i]$ is a collection of paths.*

Proof. We establish the proof by contradiction. Let $H = G[L_i]$. If H is not a collection of paths, there must exists either (1) a vertex v with degree at least 3 in H, or (2) a cycle in H. Let $v_1, v_2, \ldots$ be the clockwise ordering of vertices from H in an outerplanar embedding of G. In case (1), the vertex v connects to at least three other vertices $x, y, z \in L_i$. In this case, by Fact B.7 we know that $\{r, v, x, y, z\}$ is a $K_{2,3}$ minor, which is impossible since G is outerplanar. Similarly, in case (2), a cycle has at least three vertices. Consider any three vertices x, y, z within the same cycle of H. This implies that $\{r, x, y, z\}$ is a K_4 minor of G, a contradiction. □

Lemma B.9. *Let $G_i = G[L_i \cup L_{i+1} \cup \cdots \cup L_d]$ and let $k \in \mathbb{Z}_{\geq 0}$ be a non-negative integer. For each vertex $v \in L_i$, the set $N_{G_i,k}(v) := \{x \in L_i \mid \mathrm{dist}_{G_i}(v, x) \leq k\}$ has at most $2^k + 1$ vertices.*

Proof. We prove by induction on k. When $k = 0$ the statement is trivially true. Now we take $k = 1$. For any i and any vertex $v \in L_i$, denote by X the neighbors of v in G_i. We claim $|X| \leq 2$. Indeed, if $|X| \geq 3$ then there must be a claw minor from v to $\{x, y, z\} \subseteq X$ in G_i. This implies that $\{v, x, y, z, r\}$ is a $K_{2,3}$ minor in G, a contradiction to Fact B.6. This immediately implies that $|N_{G_i,k}(v)| \leq 2$ as well.

Suppose now that the statement is correct up to $k - 1 \geq 1$. Consider any vertex $v \in L_i$. By Lemma B.8, $G[L_i]$ is a collection of paths. Let X be the neighbor of v in L_i. Consider any path P starting from v and uses vertices in $L_{i+1} \cup L_{i+2} \cup \cdots \cup L_d$ as internal vertices and ending at any vertex on $L_i \setminus X$. We note that these paths will have lengths at least 2. Let $Y \subseteq L_i$ be all possible ending vertices of such path P. We claim that $|X| + |Y| \leq 2$. Indeed, if otherwise $|X| + |Y| \geq 3$, then $\{v\} \cup X \cup Y$ is clearly containing a $K_{1,3}$ minor. We now observe that

$$N_{G_i,k}(v) \subseteq \left(\bigcup_{x \in X} N_{G_i,k-1}(x) \right) \cup \left(\bigcup_{y \in Y} N_{G_i,k-2}(y) \right). \tag{13}$$

By induction hypothesis, for all $0 \le k' < k$ and for all $x \in X \cup Y$, $|N_{G_i,k'}(x)| \le 2^{k'} + 1$. To apply the induction, it suffices to consider several cases:

- If $|X \cup Y| = 0$, then $|N_{G_i,k}(v)| = 1$.
- If $|X \cup Y| = 1$, then $|N_{G_i,k}(v)| \le 2^{k-1} + 1 < 2^k + 1$.
- If $|X| = 2$ (which implies $|Y| = 0$), we notice that $k - 1 \ge 1$ and thus v is included in every set $N_{G_i,k-1}(x)$ with $x \in X$. Then, $|N_{G_i,k}(v)| \le 2(2^{k-1} + 1) - 1 = 2^k + 1$. Here we need to subtract 1 as v is double counted.
- If $|X \cup Y| = 2$ and $|Y| \ge 1$, then using (13), we have $|N_{G_i,k}(v)| \le (2^{k-1}+1)+(2^{k-2}+1) \le 2^k+1$.

□

Lemma B.10. *For any i, L_i can be partitioned into $O(1)$ sets $S_{i,1}, S_{i,2}, \ldots, S_{i,\ell}$ such that $S_{i,j}$ is a 5-hop independent set in the subgraph $G_i = G[L_i \cup L_{i+1} \cup \cdots \cup L_d]$. Specifically, L_i is $(17, 5)$-scattered in G_i.*

Proof. We consider the following greedy coloring algorithm on all vertices of L_i in any order: for each vertex $v \in L_i$, assign a color to v that is different from the color of any colored vertices in $N_{G_i,4}(v)$. By Lemma B.9, $2^4 + 1 = 17$ colors suffices for coloring all vertices in L_i. Observe that any two vertices of the same color are at least 5-hops away from each other. Hence, we are able to partition L_i into at most $17 = O(1)$ sets, where each set $S_{i,j}$, comprised of all vertices of color j, is a 5-hop independent set in G_i. □

Before we give the proof of Lemma 4.10, we introduce a helpful observation for the analysis.

Lemma B.11. *For any vertex $v \in L_{i+1}$, there are at most two neighbors of v in L_i in G.*

Proof. If $v \in L_{i+1}$ has at least three neighbors $x, y, z \in L_i$, we know that $\{r, x, y, z, v\}$ form a $K_{2,3}$ minor, which is a contradiction. □

Now we are ready to finish the proof of Lemma 4.10.

Proof of Lemma 4.10. The proof will be similar to the proof of Lemma 4.6, but without the recursion. We first apply the following charging argument, where we charge the cost $x_A x_B$ of each pair of intersecting districts $A \cap B \ne \emptyset$ to any vertex $v \in (A \cap B) \cup \{c_A, c_B\}$ with the smallest possible distance to the chosen root r on G. Recall that c_A and c_B are the center vertices of A and B respectively.

Consider a vertex v in the i-th BFS layer L_i. Let $Q(v)$ be the collection of district pairs charged to v. By the way we charge the pairs, we know that the total cost for v can be bounded by:

$$\mathrm{cost}(v) := \sum_{(A,B) \in Q(v)} x_A x_B \le 2 \cdot \sum_{S:\ c_S \in N_{G_i,2}(v)} x_S. \tag{14}$$

Now, it suffices to bound $\sum_{v \in V} \mathrm{cost}(v)$. Consider a district S with the center vertex $c_S \in L_i$. By (14), x_S is never involved in any $\mathrm{cost}(v)$ with $v \in L_{i+1} \cup L_{i+2} \cup \cdots \cup L_d$. We observe that by Lemma B.11, x_S is counted by at most 4 times in level $i-2$, at most 2 times in level $i-1$. Finally, we would like to analyze the number of occurrences of x_S in $\mathrm{cost}(v)$ for v in level i. By Lemma B.10, L_i can be partitioned into at most 17 5-hop independent sets $S_{i,1}, S_{i,2}, \ldots, S_{i,17}$. Since each set $S_{i,j}$ is a 5-hop independent set, x_S can only be involved in at most one vertex. This implies that x_S is added by at most 17 times in level i. Therefore, we conclude that $\sum_{v \in V} \mathrm{cost}(v) \le 2 \cdot (17 + 4 + 2) \sum_S x_S = O(1) \cdot \sum_S x_S$ as desired. □

B.8 Trees

On trees, we are able to bound the product terms for overlapping districts with just the sum of all variables.

Lemma B.12. *Let G be a tree. Suppose that $\{x_S\}$ are primal variables obtained by Theorem 6. Then, the sum of products of primal variables among all (unordered) pairs of overlapping districts $\{A, B\}$ satisfies $\sum_{A\cap B\neq\emptyset} x_A x_B \leq \sum x_S$.*

Proof. For any district S we denote c_S as their center vertex. Fix any vertex r on G as the root of the tree. For each vertex $v \in V \setminus \{r\}$, we let $\text{parent}(v)$ to be the parent vertex of v. For brevity we also define $\text{parent}(r) = \perp$. Then, each district S can be of two types: either $\text{parent}(c_S) \notin S$ (denoted as **type I**) or $\text{parent}(c_S) \in S$ (denoted as **type II**). For each district S we also define $\text{highest}(S)$ to be the vertex in S that is closest to the root on G.

Since the pairs of overlapping districts are unordered, it suffices to consider for each district A, *only* the overlapping districts B such that either (1) c_A is NOT an ancestor of c_B, or (2) $c_A = c_B$ and $\text{highest}(A) \neq \text{parent}(\text{highest}(B))$ — this condition excludes the situation where A is type II but B is type I.

Let $\mathcal{S}_A$ be the collection of all overlapping districts B that satisfies the above criteria. We can now express the sum of products as:

$$\sum_{A\cap B\neq\emptyset} x_A x_B \leq \sum_{A\in\mathcal{S}} x_A \cdot \left(\sum_{B\in\mathcal{S}_A} x_B\right) \tag{15}$$

We note that some overlapping pairs (A, B) may be double counted in the above notation so we may have inequality. For example, when both A and B are type II and c_A, c_B are siblings. Another example is when $c_A = c_B$ and A, B are both type I. Now, for each $A \in \mathcal{S}$. Depending on whether A is type I or type II, we have two cases:

- A **is type I.** In this case, by the definition of $\mathcal{S}_A$, we know that *every* district in $\mathcal{S}_A$ contains c_A. Therefore, by the primal constraints we have $\sum_{B\in\mathcal{S}_A} x_B \leq 1$.
- A **is type II.** In this case, by the definition of $\mathcal{S}_A$, we know that all districts in $\mathcal{S}_A$ contains $\text{parent}(c_A)$: whenever $c_B = c_A$, the criteria (2) guarantees that B is of type II too. Hence, using the fact that G is a tree, $\text{parent}(c_A) = \text{parent}(c_B) \in B$. Thus, we also have $\sum_{B\in\mathcal{S}_A} x_B \leq 1$ as well.

Lemma B.12 immediately follows Equation (15) and the above case analysis. ☐

By setting $\tau = 2$ in Lemma B.5, we conclude that the algorithm produces $(4 + \varepsilon)$-approximate solutions on trees.

B.9 General Graphs

Lemma 4.2. *Let G be any graph. Let $\{x_S\}$ be any feasible solution to the linear program. Then,*

$$\sum_{A,B:\ A\cap B\neq\emptyset} x_A x_B \leq \sqrt{n} \cdot \sum_S x_S.$$

Proof. We split the set $\mathcal{S}$ into two parts $\mathcal{S}_{\text{large}} := \{S \in \mathcal{S} \mid |S| \geq \sqrt{n}\}$ and $\mathcal{S}_{\text{small}} := \{S \in \mathcal{S} \mid |S| < \sqrt{n}\}$ according to the cardinality of each c-balanced district. We first claim that

$$\sum_{A,B\in\mathcal{S}_{\text{large}}:\ A\cap B\neq\emptyset} x_A x_B \leq \sqrt{n} \cdot \sum_{S\in\mathcal{S}_{\text{large}}} x_S. \tag{16}$$

As $\sum_{A,B\in\mathcal{S}_{\text{large}}} x_A x_B \leq (\sum_{S\in\mathcal{S}_{\text{large}}} x_S)^2$, it suffices to show that $\sum_{S\in\mathcal{S}_{\text{large}}} x_S \leq \sqrt{n}$. Let c_S be the center vertex of the district S. By a double counting method, we have:

$$\begin{aligned}
\sum_{S\in\mathcal{S}_{\text{large}}} x_S \cdot \sqrt{n} \leq \sum_{S\in\mathcal{S}_{\text{large}}} x_S \cdot |S| &\leq \sum_{S\in\mathcal{S}_{\text{large}}} \sum_{v\in S} x_S \leq \sum_{v\in V} \sum_{S:\ S\ni v} x_S \\
&\leq \sum_{v\in V} 1 = n. && \text{(by primal constraints of LP)}
\end{aligned}$$

Therefore Equation (16) is true and the equation captures the sums between all pair of large-size districts. For every remaining pair of districts that are overlapping, one of the districts must be in $\mathcal{S}_{\text{small}}$. For each district that overlaps with A, we charge it to one of the common vertices, arbitrarily chosen. Let's say $A \in \mathcal{S}_{\text{small}}$. Then, by partitioning all districts that overlaps with A on each vertex $v \in A$, we have

$$\sum_{B\in\mathcal{S}:\ A\cap B\neq\emptyset} x_B \leq \sum_{v\in A} \sum_{B\in\mathcal{S}:\ v\in A\cap B} x_B \leq \sum_{v\in A} 1 \leq |A|. \tag{17}$$

Finally, by combining both Equation (16) and Equation (17), we have

$$\begin{aligned}
\sum_{A,B\in\mathcal{S}:\ A\cap B\neq\emptyset} x_A x_B &\leq \sum_{A,B\in\mathcal{S}_{\text{large}}:\ A\cap B\neq\emptyset} x_A x_B + \sum_{A\in\mathcal{S}_{\text{small}},B\in\mathcal{S}:\ A\cap B\neq\emptyset} x_A x_B \\
&\leq \sqrt{n} \cdot \sum_{S\in\mathcal{S}_{\text{large}}} x_S + \sum_{A\in\mathcal{S}_{\text{small}},B\in\mathcal{S}:\ A\cap B\neq\emptyset} x_A x_B && \text{(by Equation (16))} \\
&\leq \sqrt{n} \cdot \sum_{S\in\mathcal{S}_{\text{large}}} x_S + \sum_{A\in\mathcal{S}_{\text{small}}} |A| \cdot x_A && \text{(by Equation (17))} \\
&\leq \sqrt{n} \cdot \sum_{S\in\mathcal{S}_{\text{large}}} x_S + \sum_{A\in\mathcal{S}_{\text{small}}} \sqrt{n} \cdot x_A \\
&\leq \sqrt{n} \cdot \sum_{S\in\mathcal{S}} x_S
\end{aligned}$$

as desired. □

B.10 Large Rounding Gap Examples

We first show that it is possible for the analysis reporting a rounding gap greater than 1, even if the graph is planar. We remark that, however, these examples do not imply hardness of approximation results, as it is still possible for the rounding algorithm (e.g., by testing with a smaller τ value) returning a close-to-optimal solution.

B.10.1 Constant > 1 Rounding Gap For Planar Graphs

We examine the graphs which correspond to the following two grid patterns: the (square) grid and the triangular grid.

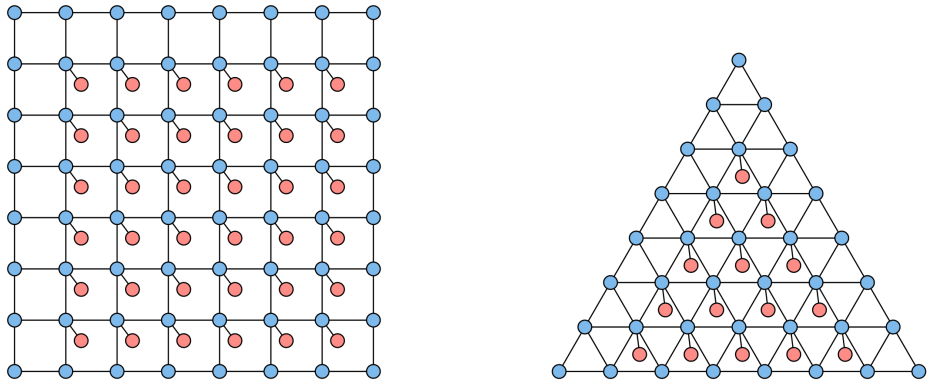

Figure 3: Construction of two types of the grid. The boundary of grid contains $O(\sqrt{n})$ vertices. Each internal (non-boundary) vertex is attached with an additional vertex for fulfilling the balancedness constraint. The weights are set as follows. (a) For the square grid construction, each grid vertex has $p_1(v) = 1$ and each attached vertex has $p_2(v) = (c-1) \times 5$. (b) For the triangular grid construction, each grid vertex has $p_1(v) = 1$ and each attached vertex has $p_2(v) = (c-1) \times 7$.

Square Grid. Consider a $\sqrt{n} \times \sqrt{n}$ square grid graph. Each grid vertex has weight $p_1(v) = 1$ and $p_2(v) = 0$. Each internal vertex is attached with an additional vertex (called a dangling vertex), with weight $p_1(v) = 0$ and $p_2(v) = (c-1) \times 5$. See the left figure in Figure 3. This assignment ensures that any c-balanced *star* district S has a center at an internal vertex and it must take all 5 neighbors (4 neighbors on the grid and one dangling vertex). An integral solution can be done by greedily tiling "cross shaped" tiles on the grid. This achieves a total weight $(1-o(1))cn$. On the other hand, we can set dual variables to be $y_v = c$ for all v on the grid, and $y_v = 0$ for all vertex v that is dangling. This ensures a feasible dual solution with objective value cn. Thus, the integrality gap of the formulated LP is at most $1 + o(1)$.

Now, we examine the rounding algorithm with a possible output from LP. A close-to-optimal fractional solution can be achieved by setting $x_S = 1/5$ for all district S. The objective value is then $(\sqrt{n}-2)^2 \cdot \frac{1}{5} \cdot (5(c-1)+5) = (1-o(1))cn$. The sum of products between overlapping districts is at least

$$\frac{1}{2} \cdot (\sqrt{n}-6)^2 \cdot 12 \cdot \frac{1}{5^2} = (1-o(1)) \cdot \frac{6}{25} \cdot n \ .$$

Since the sum of all x_S is at most $n/5 = \frac{5}{25} \cdot n$, we know that $\tau \geq 6/5$ must be set in order to apply Lemma B.5.

Triangular Grid. Consider a triangular grid where each side has $O(\sqrt{n})$ vertices. Similar to the square grid construction, we assign each grid vertex a weight $p_1(v) = 1$ and $p_2(v) = 0$. For each internal vertex, there is an additional dangling vertex attached with weight $p_1(v) = 0$ and $p_2(v) = (c-1) \times 7$. See the right figure in Figure 3. A similar argument to the square grid analysis shows that the integrality gap is also at most $1 + o(1)$. Now, a fractional solution can be achieved by setting $x_S = 1/7$ for all c-balanced star district S. The sum of products between overlapping districts is at least

$$\frac{1}{2} \cdot (1-o(1))n \cdot 18 \cdot \frac{1}{7^2} = (1-o(1)) \cdot \frac{9}{49} \cdot n \ .$$

Since the sum of all x_S is at most $n/7 = \frac{7}{49} \cdot n$, we know that $\tau \geq 9/7$ must be set in order to apply Lemma B.5. We notice that $9/7 > 6/5$, which leads to a slightly larger gap comparing with the square grid case.

B.10.2 Large Rounding Gap on General Graphs

In Section 4.4 we have shown an example reduced from hypergraph matching with both the integrality gap and the rounding gap are $\Theta(\sqrt{n})$. Here we show an instance with $O(1)$ integrality gap yet still $\Omega(\sqrt{n})$ rounding gap.

The Construction. In this construction we assume $\sqrt{n}$ to be an integer. We construct a bipartite graph $G = ((A \cup B) \cup R, E)$ with $3n$ vertices ($|A| = |B| = |R| = n$) as follows. We regard vertices in R as a $\sqrt{n} \times \sqrt{n}$ grid. Each vertex in R can be labelled as $R_{i,j}$ where $1 \leq i, j \leq \sqrt{n}$. Both A and B are partitioned into $\sqrt{n}$ sets $\{A_i\}$ and $\{B_j\}$, each containing $\sqrt{n}$ vertices. For each i, each vertex $v \in A_i$ has incident edges to all vertices $R_{i,1}, R_{i,2}, \ldots, R_{i,n}$. For each j, each vertex $v \in B_j$ has incident edges to all vertices $R_{1,j}, R_{2,j}, \ldots, R_{n,j}$. The degree of each vertex in A and B is $\sqrt{n}$ and the degree of each vertex in R is then $2\sqrt{n}$.

Now, we assign the weights. Each vertex v in $A \cup B$ has weight $p_1(v) = (c-1)\sqrt{n}$ and $p_2(v) = 0$. Each vertex v in R has weight $p_1(v) = 0$ and $p_2(v) = 1$. With this weight assignment, the only c-balanced star districts that can be formed are the districts that are centered at any vertex $v \in A \cup B$ and containing all neighbors of v. There are exactly $2n$ districts.

A Fractional Solution. For each $v \in A \cup B$ we denote S_v to be the (unique) district centered at v. A fraction solution for the formulated LP can be simply setting $x_{S_v} = 1/(2\sqrt{n})$. This is definitely feasible since the degree of vertices in R is $2\sqrt{n}$. This solution is also optimal since the weight contribution of p_2 to the objective function is maximized.

Integrality Gap. It is also easy to see that the integrality gap is 1: for each set A_i we pick an arbitrary vertex $v \in A_i$ and add S_v to the final districting. The final districting covers all vertices in B so it is an optimal solution as well.

Rounding Gap. We now show that this example gives an $\Omega(\sqrt{n})$ rounding gap. Indeed, for each vertex $v \in A$, the district S_v intersects with *all* S_u where $u \in B$: suppose $v \in A_i$ and $u \in B_j$, then they share a common vertex $R_{i,j} \in S_v \cap S_u$. Therefore, we have

$$\sum_{S,T:\ S\cap T\neq\emptyset} x_S x_T \geq |A| \cdot |B| \cdot \left(\frac{1}{2\sqrt{n}}\right)^2 = \frac{n}{4}.$$

On the other hand, the sum of all primal variables is

$$\sum_S x_S = 2n \cdot \left(\frac{1}{2\sqrt{n}}\right) = \sqrt{n}.$$

Therefore, the rounding gap is $\sqrt{n}/4 = \Omega(\sqrt{n})$.

B.10.3 Large Rounding Gap for Deterministic Greedy Rounding

In this section, we give an instance on which deterministic greedy rounding fails to give a good approximation, motivating the need for randomized rounding. Here in the deterministic greedy rounding, we sort the primal variables in decreasing order and choose a district as long as it does not overlap with any chosen districts.

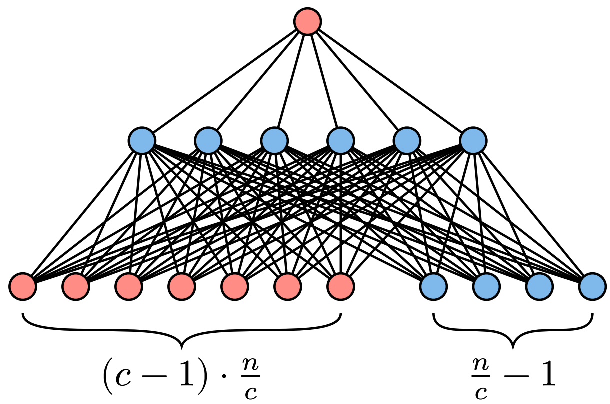

Figure 4: Construction of hard instance for greedy rounding.

The construction: Refer to Figure 4 for the construction. The graph can be considered as divided into three layers, with single vertex in first layer (denoted henceforth by v), $c-1$ vertices in second layer (call this set X) and $n-1$ vertices in the third layer (call this set Y). v has p_1 population 1, each vertex in X has p_2 population 1, Y has $(c-1)\frac{n}{c}$ vertices with p_1 population 1 and $\frac{n}{c}-1$ vertices with p_2 population 1. Vertex v is connected to every vertex in X and there exists a complete bipartite graph between X and Y. The total number of vertices in this graph is $n+c-1$.

The c-balanced districts are as follows: v and all its neighbors in X form a c-balanced district of weight c (call this district S_v). Each node $u \in X$ forms a c-balanced district with all its neighbors in Y, each of which has weight n (denoted by S_u for $u \in X$). Note that all these districts are star shaped. Consider the fractional solution, where $x(S_u) = \frac{1}{c-1}$ for each district centered at $u \in X$ and $x(S_v) = 1 - \frac{1}{c-1}$. It's straightforward to verify that this forms a feasible solution, with objective value $n + c - \frac{c}{c-1}$, which forms a $(1+\varepsilon)$-approximate solution since the optimal solution is to pick any one district centered at a vertex in X. For constant $c > 3$, we have $1 - \frac{1}{c-1} > \frac{1}{c-1}$, and as a result the greedy rounding that sorts districts by $x(S)$ values for districts S, chooses the district S_v with weight c (which is a constant), and cannot choose any other districts. This gives an $\Omega(n)$-approximation.

C Omitted Proofs from Section 5

Theorem 7. *There exists an FPTAS algorithm solving* Complete-Graph-c-Balanced-Districting *so that for all $c > 2$, $0 < \varepsilon < \frac{1}{2}\ln(c-1)$, and complete graph (V, E) of n nodes with functions of weights $\mathbf{p} = (p_1, p_2)$, the algorithm outputs an e^{ε}-approximation in $O\left(\varepsilon^{-4} n^6 (\ln w(V))^4\right)$ time where $w(V) = \sum_{v\in V} p_1(v) + p_2(v)$.*

Proof of Theorem 7. Given an arbitrary ordering on blocks, let L^i be the set of all values that can be obtained by selecting some subset of the first i blocks $\{v_1, \ldots, v_i\}$,

$$L^i = \left\{ \exists S \subseteq [i], \sum_{j \in S} \mathbf{p}(v_j) \right\} \subset \mathbb{Z}^2_{\geq 0}.$$

We use induction to show that L^i_1 in Algorithm 1 is an $(\ell_1, \frac{\varepsilon i}{n})$-trimmed of L^i for all i. The base case $i = 0$ is trivially holds as $L^0 = \emptyset$. Suppose L^{i-1}_1 is an $\left(\ell_1, \frac{\varepsilon(i-1)}{n}\right)$-trimmed of L^{i-1}. For any

$\mathbf{q}+\mathbf{p}(v_i) \in L^i \setminus L^{i-1}$ with $\mathbf{q} \in L^{i-1}$, by the induction hypothesis, there exists $\mathbf{q}' \in L_1^{i-1}$ which $\frac{\varepsilon(i-1)}{n}$-approximates and ℓ_1-dominates $\mathbf{q}$. Because $p_1(v_i), p_2(v_i) \geq 0$,

$$\frac{q_1'+p_1(v_i)}{q_1+p_1(v_i)}, \frac{q_2'+p_2(v_i)}{q_2+p_2(v_i)} \in [e^{\frac{-\varepsilon(i-1)}{n}}, e^{\frac{\varepsilon(i-1)}{n}}] \text{ and } \ell_1(\mathbf{q}'+\mathbf{p}(v_i)) \geq \ell_1(\mathbf{q}+\mathbf{p}(v_i)).$$

On the other hand, by the definition of L_1^i, for any $\mathbf{q}'+\mathbf{p}(v_i) \in L_1^{i-1}+\mathbf{p}(v_i)$ there exists $\mathbf{q}'' \in L_1^i$ so that

$$\frac{q_1''}{q_1'+p_1(v_i)}, \frac{q_2''}{q_2'+p_2(v_i)} \in [e^{\frac{-\varepsilon}{n}}, e^{\frac{\varepsilon}{n}}] \text{ and } \ell_1(\mathbf{q}'') \geq \ell_1(\mathbf{q}'+\mathbf{p}(v_i))$$

Combining these two proves that $\mathbf{q}''$ $\frac{\varepsilon i}{n}$-approximates and ℓ_1-dominates $\mathbf{q}+\mathbf{p}(v_i)$. The identical argument holds for all $\mathbf{q} \in L^{i-1} \subseteq L^i$. Thus, we show L_1^i is an $(\ell_1, \frac{\varepsilon i}{n})$-trimmed of L^i for all i. Similar argument applies to L_2^i.

Let $\mathbf{q}^*$ be the optimal c-balanced value in L^n. Suppose $q_2^* \geq q_1^*$. Since L_1^n is (ℓ_1, ε)-trimmed to L^n, there exists $\mathbf{q}' \in L_1^n$ that ε-approximates and ℓ_1-dominates $\mathbf{q}^*$. Because $\mathbf{q}'$ ε-approximates $\mathbf{q}^*$, the approximation guarantee holds, $q_1'+q_2' \geq e^{-\varepsilon}(q_1^*+q_2^*)$. Now we show $\mathbf{q}'$ is also c-balanced. Because $\mathbf{q}^*$ is c-balanced and $\mathbf{q}'$ ℓ_1-dominates $\mathbf{q}^*$,

$$0 \leq (c-1)q_1^* - q_2^* = \ell_1(\mathbf{q}^*) \leq \ell_1(\mathbf{q}').$$

Moreover, because $q_2^* \geq q_1^*$ and $\mathbf{q}'$ ε-approximates $\mathbf{q}^*$, we have

$$(c-1)q_2' \geq (c-1)e^{-\varepsilon}q_2^* \geq (c-1)e^{-\varepsilon}q_1^* \geq (c-1)e^{-2\varepsilon}q_1' \geq q_1'$$

where the last inequality holds because $\frac{1}{2}\ln(c-1) \geq \varepsilon$. Combining these two, we have $\ell_1(\mathbf{q}')$ and $\ell_2(\mathbf{q}') \geq 0$ completing the proof. Similarly, if $q_2^* \geq q_1^*$, there exists an ε-approximation and c-balanced solution in L_2^n.

The running time of i-th iteration is $O(|L_1^i|^2+|L_2^i|^2)$ which can be bounded as the following. Consider a geometric grid with vertices in $\{(e^{\frac{j}{n}\varepsilon}, e^{\frac{k}{n}\varepsilon}) : j,k = 0,\dots,\lceil \frac{n}{\varepsilon}\ln w(V)\rceil\}$. Because $L_1^i \subseteq [w(V)]^2$ and no two points in can be in a same rectangle after trimming, the size of L_1^i is bounded by the size of grid $O(\frac{n^2}{\varepsilon^2}(\ln w(V))^2)$. Therefore, the running time of Algorithm 1 is $O(\frac{n^5}{\varepsilon^4}(\ln w(V))^4)$. The additional n in the theorem statement is to reconstruct the set. □

Theorem 8. *There exists an FPTAS for the c-balanced districting problem on trees. That is, for all $c > 2$, $0 < \varepsilon < \frac{1}{2}\ln(c-1)$, and tree graph (V,E) of n blocks with population functions $\mathbf{p}$, the algorithm outputs an e^ε-approximation in $O\left(\varepsilon^{-6}n^8(\ln w(V))^6\right)$ time.*

Analogous to Section 5.1, with slight abuse of notation, let $\ell_1(\mathbf{s}) = (c-1)s_1 - s_2$ and $\ell_2(\mathbf{s}) = (c-1)s_2 - s_1$ for all $\mathbf{s} \in \mathbb{R}^3$. Given $j = 1,2$, $\varepsilon \geq 0$, and $\mathbf{s}, \mathbf{s}'$, $\mathbf{s}$ is ℓ_j-*dominated* by $\mathbf{s}'$ if $\ell_j(\mathbf{s}) \leq \ell_j(\mathbf{s}')$, and $\mathbf{s}$ is an ε-*approximate* of $\mathbf{s}'$ if $s_1/s_1', s_2/s_2', s_3/s_3' \in [e^{-\varepsilon}, e^\varepsilon]$ with $0/0 := 1$. Finally, given two sets $L, L' \subset \mathbb{R}^3$, L' is a (ℓ_j, ε)-trimmed of L if $L' \subseteq L$ and for all $\mathbf{s} \in L$ there exists $\mathbf{s}' \in L'$ which is an ε-approximate and ℓ_j-dominates $\mathbf{s}$, and $L + L' = \{\mathbf{s}+\mathbf{s}' : \mathbf{s} \in L \text{ and } \mathbf{s}' \in L'\}$.

The following lemma shows that the parameter ε decay smoothly under composition and addition.

Lemma C.1. *Given $\varepsilon, \varepsilon_1, \varepsilon_2 \geq 0$ and $j = 1,2$, if L_1' is an (ℓ_j, ε_1)-trimmed of L_1 and L_2' is an (ℓ_j, ε_2)-trimmed of L_2, $L'' =$ `Trim`$(L_1'+L_2', \ell_j, \varepsilon)$ is a $(\ell_j, \varepsilon + \max(\varepsilon_1, \varepsilon_2))$-trimmed of $L_1 + L_2$. Moreover, if $L_1 + L_2 \subseteq \{0,1,\dots,W\}^3$ for some $W \in \mathbb{Z}_{>0}$, $|L''| \leq \left(1+\frac{\ln W}{\varepsilon}\right)^3$.*

Proof. For the first part, $L'_1 + L'_2 \subseteq L_1 + L_2$ is trivial. For any $\mathbf{s}_1 = (s_{1,1}, s_{1,2}, s_{1,3}) \in L_1$ and $\mathbf{s}_2 = (s_{2,1}, s_{2,2}, s_{2,3}) \in L_2$, there exists $\mathbf{s}'_1 \in L'_1$ and $\mathbf{s}'_2 \in L'_2$ so that $\ell_j(\mathbf{s}'_1) \geq \ell_j(\mathbf{s}_1), \ell_j(\mathbf{s}'_2) \geq \ell_j(\mathbf{s}_2)$,

$$\frac{s_{1,1}}{s'_{1,1}}, \frac{s_{1,2}}{s'_{1,2}}, \frac{s_{1,3}}{s'_{1,3}} \in [e^{-\varepsilon_1}, e^{\varepsilon_1}], \text{ and } \frac{s_{2,1}}{s'_{2,1}}, \frac{s_{2,2}}{s'_{2,2}}, \frac{s_{2,3}}{s'_{2,3}} \in [e^{-\varepsilon_2}, e^{\varepsilon_2}].$$

Hence, $\mathbf{s}'_1 + \mathbf{s}'_2 \in L'_1 + L'_2$ and $\mathbf{s}_1 + \mathbf{s}_2 \in L_1 + L_2$ satisfy $\ell_1(\mathbf{s}'_1 + \mathbf{s}'_2) \geq \ell_1(\mathbf{s}_1 + \mathbf{s}_2)$,

$$\frac{s_{1,1} + s_{2,1}}{s'_{1,1} + s'_{2,1}}, \frac{s_{1,2} + s_{2,2}}{s'_{1,2} + s'_{2,2}}, \frac{s_{1,3} + s_{2,3}}{s'_{1,3} + s'_{2,3}} \in [e^{-\max \varepsilon_1, \varepsilon_2} e^{\max \varepsilon_1, \varepsilon_2}].$$

Because L'' is (ℓ_1, ε)-trimmed of $L'_1 + L'_2$, $L'' \subset L'_1 + L'_2 \subset L_1 + L_2$, and L'' is $(\ell_1, \varepsilon + \max(\varepsilon_1, \varepsilon_2))$-trimmed of $L_1 + L_2$ using a similar argument.

For the second part, because $L'' \subseteq L_1 + L_2$, the value of L'' is in $\{0, 1, \ldots, W\}^3$. On the other hand, consider a geometric grid with vertices in $\{\mathbf{s} \in [0, V]^3 : s_1, s_2, s_3 = 0, 1, e^{\varepsilon}, e^{2\varepsilon}, \ldots, \}$. As no two points in L'' can be in the same rectangle, the size of L'' is bounded by the size of the grid, $O((\varepsilon^{-1} \ln W)^3)$. □

Proof of Theorem 8. Note that the size of an optimal solution $\mathcal{T}^*$ is $s_3^{root}(\mathcal{T}^*)$, and $(0, 0, s_3^{root}(\mathcal{T}^*)) \in L^{root}$. If L_1^{root} and L_2^{root} are (ℓ_1, ε)-trimmed and (ℓ_2, ε)-trimmed of L^{root} respectively, there exists $\mathbf{s} \in L_1^{root} \subseteq L^{root}$ such that $s_1 = s_2 = 0$ and $s_3 \geq e^{-\varepsilon} s_3^{root}(\mathcal{T}^*)$ which yields an approximation ratio of e^{ε} and completes the proof. To this end, let $size(v)$ denote the number of block in the subtree of v including v and $\varepsilon_v := size(v)\frac{\varepsilon}{n} \leq \varepsilon$ with $\varepsilon'_v = (size(v) - 1)\frac{\varepsilon}{n}$. We will use induction. As we cannot decide whether the current incomplete district is c-balanced or not in `grow`, our induction will be on slightly extended sets instead of L^v, defined as following

$$\bar{L}^v = \left\{ \mathbf{s}^v(\mathcal{T}) : \begin{matrix} \mathcal{T} = (T_1, \ldots, T_m) \text{ and for all } i, T_i \text{ is } c\text{-balanced} \\ \text{if it is fully contained in the subtree of } v. \end{matrix} \right\} \tag{18}$$

where $L^v \subseteq \bar{L}^v$ and $L^{root} = \bar{L}^{root}$ because $\bar{L}^v$ allows $\mathcal{T}$ to contain unbalanced districts outside of the subtree of v. Additionally, we partition $\bar{L}^v$ into three sets:

$$\begin{aligned} I^v &= \{\mathbf{s}^v(\mathcal{T}) \in \bar{L}^v : v \text{ is incomplete for some } T \in \mathcal{T}.\} \\ C^v &= \{\mathbf{s}^v(\mathcal{T}) \in \bar{L}^v : v \text{ is consolidating.}\} \\ A^v &= \{\mathbf{s}^v(\mathcal{T}) \in \bar{L}^v : v \text{ is absent.}\} \end{aligned}$$

We use induction to show that L_1^v is a (ℓ_1, ε_v)-trimmed of $\bar{L}^v$ and L_2^v is a (ℓ_1, ε_v)-trimmed of $\bar{L}^v$ for all v.

When v is a leaf, the statement holds. Now suppose v is not the root and for all v's child u, L_1^u is a (ℓ_1, ε_u)-trimmed of $\bar{L}^u$ and L_2^u is a (ℓ_2, ε_u)-trimmed of $\bar{L}^u$.

First, because I^v can be written as

$$I^v = (p_1(v), p_2(v), 0) + \sum_{\text{child } u} \bar{L}^u,$$

and I_1 is $(p_1(v), p_2(v), 0) + \sum_{\text{child } u} L_1^u$ applied `Trim` $deg(v) - 1$ times. Thus I_1 (and I_2) is an (ℓ_1, ε'_v)-trimmed (and (ℓ_2, ε'_v)-trimmed) of I^v, due to Lemma C.1 and $\max_u \varepsilon_u + \frac{\varepsilon}{n}(deg(v) - 1) \leq \varepsilon'_v$.

Second, we show that A is an (ℓ_1, ε'_v) and (ℓ_2, ε'_v)-trimmed of A^v. For any districting $\mathcal{T}$ satisfying the condition in Equation (18), if v is absent in $\mathcal{T}$ so that $\mathbf{s}^v(\mathcal{T}) \in A^v$, we can create a new districting $\mathcal{T}' = \mathcal{T} \cup \{v\}$ so that v is incomplete in $\mathcal{T}'$ and thus $\mathbf{s}^v(\mathcal{T}') \in I^v$. Hence

$$s_1^v(\mathcal{T}) = s_2^v(\mathcal{T}) = 0 \text{ and } s_3^v(\mathcal{T}) = s_3^v(\mathcal{T}').$$

Algorithm 5: FPTAS on tree graphs

Input: $\varepsilon > 0$, $c > 2$, a rooted tree (V, E), and population functions $\mathbf{p} = (p_1, p_2)$

Output: The size of optimal c-balanced districts

Function `Trim`(L, ℓ, ε)**:**

  Sort $L = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\}$ so that $\ell(\mathbf{s}_1) \geq \ell(\mathbf{s}_2) \geq \cdots \geq \ell(\mathbf{s}_m)$;

  Set $L_{out} = \emptyset$ and $L_{rem} = \emptyset$;

  for $i = 1, \ldots, m$ **do**

    if $\mathbf{s}_i \notin L_{rem}$ **then**

      $L_{out} \leftarrow L_{out} \cup \{\mathbf{s}_i\}$;

      for $j = i + 1, \ldots, m$ **do**

        if $\mathbf{s}_i$ *ε-approximates* $\mathbf{s}_j$ **then**

          $L_{rem} \leftarrow L_{rem} \cup \{\mathbf{s}_j\}$

  return L_{out};

Function `grow`(v, ε)**:**

  Set $L_1^v, L_2^v \leftarrow \emptyset$;

  Set $I_1, I_2 \leftarrow \{(p_1(v), p_2(v), 0)\}$ `// Only include itself`

  for *Each v's child u* **do**

    $(L_1^u, L_2^u) \leftarrow$ `grow`(u, ε);

    $I_1 \leftarrow$ `Trim`$(I_1 + L_1^u, \ell_1, \varepsilon)$;

    $I_2 \leftarrow$ `Trim`$(I_2 + L_2^u, \ell_2, \varepsilon)$;

  $C \leftarrow \{(0, 0, s_1 + s_2 + s_3) : \exists \mathbf{s} \in I_1 \cup I_2, \ell_1(\mathbf{s}), \ell_2(\mathbf{s}) \geq 0\}$;

  $A \leftarrow \{(0, 0, s_3) : \exists \mathbf{s} \in I_1 \cup I_2\}$;

  if $v \neq root$ **then**

    $L_1^v \leftarrow$ `Trim`$(I_1 \cup C \cup A, \ell_1, \varepsilon)$;

    $L_2^v \leftarrow$ `Trim`$(I_2 \cup C \cup A, \ell_2, \varepsilon)$;

  else

    $L_1^v \leftarrow$ `Trim`$(C \cup A, \ell_1, \varepsilon)$;

    $L_2^v \leftarrow$ `Trim`$(C \cup A, \ell_2, \varepsilon)$;

  return (L_1^v, L_2^v);

$L_1^{root}, L_2^{root} \leftarrow$ `grow`$(root, \varepsilon/n)$;

return $\max\{s_3 : \mathbf{s} \in L_1^{root}, L_2^{root}\}$

Because I_1 is an (ℓ_1, ε'_v)-trimmed of I^v, $I_1 \subseteq I^v$ and I_1 has $\mathbf{z}' = (z'_1, z'_2, z'_3)$ so that ε'_v-approximates $\mathbf{s}^v(\mathcal{T}')$ so that

$$e^{-\varepsilon'_v} \leq \frac{z'_3}{s^v_3(\mathcal{T}')} \leq e^{\varepsilon'_v}.$$

By the definition of A in Algorithm 5, there exists $\mathbf{z} \in A$ so that $z_1 = z_2 = 0$ and $z_3 = z'_3$. Combining these we have

$$s^v_1(\mathcal{T}) = z_1 = s^v_2(\mathcal{T}) = z_2 = 0 \text{ and } e^{-\varepsilon'_v} \leq \frac{z_3}{s^v_3(\mathcal{T})} \leq e^{\varepsilon'_v}.$$

Because $I_1, I_2 \subseteq I^v$, $A \subseteq A^v$.

Third, we show that C is an (ℓ_1, ε'_v) and (ℓ_2, ε'_v)-trimmed of C^v. If v is consolidating for $T \in \mathcal{T}$, we creates $\mathcal{T}'$ so that v is incomplete by extending T outside the subtree of v and thus $\mathbf{s}^v(\mathcal{T}') \in I^v$. Note that $s^v_1(\mathcal{T}) = s^v_2(\mathcal{T}) = 0$, $s^v_1(\mathcal{T}') = p_1(T)$, $s^v_2(\mathcal{T}') = p_2(T)$ and

$$s^v_3(\mathcal{T}) = s^v_1(\mathcal{T}') + s^v_2(\mathcal{T}') + s^v_3(\mathcal{T}').$$

If $p_1(T) \leq p_2(T)$, because I_1 is an (ℓ_1, ε'_v)-trimmed of I^v, there exists $\mathbf{z}' \in I_1$ which ε'_v-approximates and ℓ_1-dominates $\mathbf{s}(\mathcal{T}')$. Thus, $z'_1 + z'_2 + z'_3 \geq e^{\varepsilon'_v}\left(s_1(\mathcal{T}') + s_2(\mathcal{T}') + s_3(\mathcal{T}')\right)$. Now we show that (z'_1, z'_2) is c-balanced, so $(0, 0, z'_1 + z'_2 + z'_3) \in C$. Because $\mathbf{z}'$ ℓ_1-dominates $\mathbf{s}(\mathcal{T}')$ $\ell_1(\mathbf{z}') \geq \ell_1(\mathbf{s}(\mathcal{T}')) \geq 0$. Because $p_1(T) \leq p_2(T)$, we have

$$(c-1)z'_2 \geq (c-1)e^{-\varepsilon'_v} s_2(T) \geq (c-1)e^{-\varepsilon'_v} s_1(T) \geq (c-1)e^{-2\varepsilon'_v} z'_1 \geq z'_1$$

because $(c-1) \geq e^{2\varepsilon} \geq e^{2\varepsilon'_v}$. Therefore, $(0, 0, z'_1 + z'_2 + z'_3) \in C$ ε'_v-approximates, ℓ_1 and ℓ_2 dominates $\mathbf{s}^v(\mathcal{T}) \in C^v$. Similarly if $p_1(T) \geq p_2(T)$, there exists $\mathbf{z}'' \in I_2$ so that $(0, 0, z''_1 + z''_2 + z''_3) \in C$ ε'_v-approximates, ℓ_1 and ℓ_2-dominates $\mathbf{s}(\mathcal{T})$.

Combining these three cases, $L^v_1 = \texttt{Trim}(I_1 \cup A \cup C, \ell_1, \varepsilon/n)$ is an (ℓ_1, ε_v)-trimmed of $\bar{L}^v$ because $\varepsilon'_v + \varepsilon/n = \varepsilon_v$. The argument for $v = \textit{root}$ is similar.

For time complexity, by Lemma C.1, the size of I_1, I_2, L^v_1, L^v_2 and thus A and C are bounded by $O\left((\frac{n \ln W}{\varepsilon})^3\right)$ with $W = w(V)$. Each trimming process $\texttt{Trim}$ takes quadratic time in the size, and there are n recursive calls of $\texttt{grow}$, so the running time is bounded by $O(n(\frac{n \ln W}{\varepsilon})^6)$. The additional n in the theorem statement is to reconstruct the districting. □