Optimal LP Rounding and Linear-Time Approximation Algorithms for Clustering Edge-Colored Hypergraphs

Nate Veldt [1]

Abstract

We study the approximability of an existing framework for clustering edge-colored hypergraphs, which is closely related to chromatic correlation clustering and is motivated by machine learning and data mining applications where the goal is to cluster a set of objects based on multiway interactions of different *categories* or *types*. We present improved approximation guarantees based on linear programming, and show they are tight by proving a matching integrality gap. Our results also include new approximation hardness results, a combinatorial 2-approximation whose runtime is linear in the hypergraph size, and several new connections to well-studied objectives such as vertex cover and hypergraph multiway cut.

1. Introduction

Partitioning a graph into well-connected clusters is a fundamental algorithmic problem in machine learning. A recent focus in the machine learning community has been to develop algorithms for clustering problems defined over data structures that generalize graphs and capture rich information and metadata beyond just pairwise relationships. One direction has been to develop algorithms for clustering and learning over hypergraphs (Li & Milenkovic, 2017; 2018; Fountoulakis et al., 2021; Hein et al., 2013), which can model multiway relationships between data objects rather than just pairwise relationships. Another recent focus has been to develop techniques for clustering edge-colored graphs (and hypergraphs), where edge colors represent the *type* or *category* of pairwise interaction modeled by the edge (Amburg et al., 2020; Bonchi et al., 2015; Anava et al., 2015; Klodt et al., 2021; Xiu et al., 2022; Amburg et al., 2022). Edge color labels are relevant for many differ-

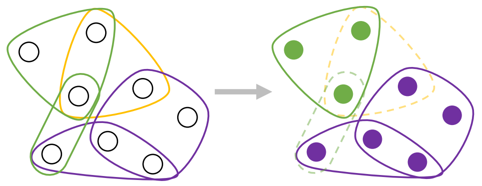

Figure 1. The best node coloring of this edge-colored hypergraph leaves three edges satisfied and two unsatisfied (dashed lines).

ent applications. In co-authorship (hyper)graphs, an edge color may represent the type of publication venue or discipline (Amburg et al., 2020; Bonchi et al., 2012), which can be used to better ascertain the academic field an author belongs to. Edges in brain networks may have color labels to indicate differences in co-activation patterns between brain regions (Crossley et al., 2013). In temporal graph analysis, edges within the same time period can be associated with the same color (Amburg et al., 2020), which provides information about how interaction patterns evolve over time. In addition to these settings, algorithms for edge-colored graphs and hypergraphs have been applied to cluster food ingredients co-appearing in different recipes (where edge colors indicate cuisine types) (Amburg et al., 2020; Klodt et al., 2021; Xiu et al., 2022), genes in biological networks (where edge colors indicate gene interaction types) (Bonchi et al., 2012), and users of social media platforms like Facebook and Twitter (where edge colors indicate types of social circles) (Klodt et al., 2021; Xiu et al., 2022).

The Edge-Colored Clustering problem. Edge-Colored Clustering (ECC) is a combinatorial optimization framework for clustering graphs and hypergraphs with edge colors. This problem is also known by other related names such as *colored clustering* or *categorical edge clustering*. Let $H = (V, E)$ be a hypergraph where each hyperedge $e \in E$ is associated with one of k colors for some $k \in \mathbb{N}$. The goal of ECC is to assign colors to *nodes* in such a way that hyperedges tend to contain nodes that all have the same color as the hyperedge (see Figure 1). This can also be described as a clustering problem where one forms a single cluster of nodes for each color, and the goal is to do so in such a way that the cluster of nodes with color c tends to contain hyperedges with color c. Note that the nodes in cluster c do not necessarily need to be connected in the hypergraph in this formulation. To defined the objective

[1]Department of Computer Science and Engineering, Texas A&M University, College Station, Texas, USA. Correspondence to: Nate Veldt <nveldt@tamu.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

more precisely, if all of the nodes in a hyperedge $e \in E$ are given the same color as e, then the hyperedge is *satisfied*, otherwise it is *unsatisfied* and we say the node color assignment has made a *mistake* at this hyperedge. The goal is then to set node colors in a way that minimizes the number (or weight) of unsatisfied edges (the MinECC objective), or to maximize the number (or weight) of satisfied edges (the MaxECC objective). These are equivalent at optimality but different from the perspective of approximations. ECC is NP-hard (Amburg et al., 2020) but permits nontrivial approximation algorithms, whose approximation factors may depend on the number of colors k and the rank r of the hypergraph (the maximum hyperedge size).

Background and previous results. Angel et al. (2016) were the first to study the problem over graphs (the $r = 2$ case), focusing on MaxECC. They showed the objective is polynomial time solvable for $k = 2$ colors but NP-hard when $k \geq 3$, and provided a $\frac{1}{e^2}$-approximation algorithm by rounding a linear programming (LP) relaxation. A sequence of follow-up papers (Ageev & Kononov, 2014; 2020; Alhamdan & Kononov, 2019) improved the best approximation factor to $4225/11664 \approx 0.3622$ (Ageev & Kononov, 2020), and showed it is NP-hard to approximate above a factor $241/249 \approx 0.972$ (Alhamdan & Kononov, 2019). All of these results apply to MaxECC and $r = 2$. Cai & Leung (2018) also focused on graphs but considered the problem from the perspective of parameterized complexity, showing that MinECC and MaxECC are fixed-parameter tractable (FPT) in the solution size.

Amburg et al. (2020) initiated the study of ECC in hypergraphs, focusing on MinECC. They gave a $\min\left\{2 - \frac{1}{k}, 2 - \frac{1}{r+1}\right\}$-approximation algorithm based on rounding an LP relaxation, and provided combinatorial algorithms with approximation factors scaling linearly in r. Finally, they showed that the problem can be reduced in an approximation-preserving way to node-weighted multiway cut (Node-MC) (Garg et al., 2004). This leads to a $2(1 - 1/k)$ approximation by rounding the Node-MC LP relaxation. Amburg et al. (2022) later extended this framework for the task of diverse group discovery and team formation. Recently, Kellerhals et al. (2023) gave improved FPT algorithms and parameterized hardness results.

Other related work. ECC is closely related to chromatic correlation clustering (Bonchi et al., 2015; Anava et al., 2015; Klodt et al., 2021; Xiu et al., 2022), an edge-colored generalization of correlation clustering (Bansal et al., 2004). The reduction to Node-MC also situates MinECC within a broad class of multiway partition problems (Ene et al., 2013; Garg et al., 2004; Dahlhaus et al., 1994; Călinescu et al., 2000; Chekuri & Ene, 2011a;b; Chekuri & Madan, 2016), such as hypergraph multiway cut (Hyper-MC). Appendix A provides a deeper discussion of related work.

Our contributions. There are many natural questions on the approximability of ECC that are unresolved in previous work, especially relating to the less studied MinECC objective. Amburg et al. (2020) provide an indirect $2(1 - 1/k)$-approximation for MinECC by rounding the Node-MC LP relaxation in a reduced graph. When $k \leq 2(r + 1)$, this is better than the approximation factor they obtain by rounding the canonical MinECC LP relaxation. An open question is to understand the exact relationship between these LP relaxations. In general, can we improve on the best LP rounding algorithms? For combinatorial algorithms, the best existing approximation factors scale linearly in r (Amburg et al., 2020). Can we design a combinatorial algorithm whose approximation factor is constant with respect to r and k? Finally, although the problem is known to be NP-hard and MaxECC is known to be APX-hard (Alhamdan & Kononov, 2019), can we say anything more about approximation hardness for MinECC?

We answer all of these open questions, and in the process prove new connections to other well-studied combinatorial objectives. In terms of linear programming algorithms:

- We prove that the MinECC LP relaxation is always at least as tight as the Node-MC LP relaxation, and in some cases is strictly tighter (Theorem 2.2).
- We improve the best approximation factors for rounding the MinECC LP relaxation from $\min\left\{2 - \frac{1}{k}, 2 - \frac{1}{r+1}\right\}$ to $\min\left\{2 - \frac{2}{k}, 2 - \frac{2}{r+1}\right\}$ (Theorems 3.1, 3.2, and 4.6).
- We prove a matching integrality gap, showing that our LP rounding schemes are in fact optimal (Lemma 2.1).

For the graph objective ($r = 2$), our approximation factor is $\frac{4}{3}$, improving on the previous best $\frac{5}{3}$-approximation (Amburg et al., 2020). Our approximation result for $r = 2$ is the most challenging and in-depth result in our paper, and relies on a new technique for using auxiliary linear programs to bound the worst-case approximation factor obtained when rounding the MinECC LP. This proof requires a careful case analysis; we also formally prove why alternative strategies for trying to round the LP relaxation (which appear potentially easier at first glance) will in fact fail to provide a $\frac{4}{3}$-approximation (Lemma 4.7). Aside from our linear programming results, our contributions include the following:

- We prove that Vertex Cover is reducible to MinECC in an approximation preserving way (Theorem 5.1), implying APX-hardness. The details of our reduction allow us to further prove UGC-hardness of approximation below $2 - (2 + o_r(1))\frac{\ln \ln r}{\ln r}$ (for arbitrarily large k). The reduction also implies hardness results for hypergraph MaxECC.
- We prove that hypergraph MinECC is reducible to Vertex Cover in an approximation preserving way (Theorem 5.2). Explicitly forming the reduced Vertex Cover

instance lead to combinatorial 2-approximation algorithms that run in time $O(\sum_{v\in V} d_v^2 + |E|^2)$.

- We develop more careful 2-approximation algorithms that run in $O(\sum_{e\in E} |e|)$, i.e., linear in the hypergraph size.

Our results improve significantly on the previous best approximation guarantees for MINECC and are also tight or near-tight with respect to different types of lower bounds. Our LP rounding scheme is optimal in that in matches the integrality gap we show. All of our algorithms for hypergraph MINECC also have asymptotically optimal approximation factors assuming the unique games conjecture, since approximating VERTEX COVER by a constant smaller than 2 is UGC-hard (Khot & Regev, 2008). Our combinatorial algorithms also have asymptotically optimal runtimes, since it takes $\Omega(\sum_{e\in E} |e|)$ time simply to read the hypergraph input. In addition to our theoretical results, we confirm in numerical experiments that our algorithms are very practical. Our empirical contributions include a new large benchmark dataset for edge-colored hypergraph clustering and a very fast method that uses additional heuristics to improve the practical performance of our combinatorial algorithms.

2. MINECC Linear Programming Framework

An instance of MINECC is given by an edge-colored hypergraph $H = (V, E, C, \ell)$ with node set V and edge set E; $w_e \geq 0$ represents the nonnegative weight for $e \in E$. We use the term *edge* even in the hypergraph setting (rather than *hyperedge*), to use uniform terminology between the graph and hypergraph setting. $C = [k] = \{1, 2, \ldots k\}$ is a set of edge colors and r denotes the maximum hyperedge size. The function $\ell \colon E \to C$ maps each edge to a color in C, and $E_c \subseteq E$ denotes the set of edges with color $c \in C$.

Let Y be a map from nodes to colors where $Y[v] \in C$ is the color of node v. For $e \in E$, if there is any node $v \in e$ such that $Y[v] \neq \ell(e)$, the map Y has made a mistake at edge e, and this incurs a penalty of w_e. This is equivalent to partitioning nodes into k clusters, with each cluster corresponding to one color, in a way that minimizes the weight of edges that are not completely contained in the cluster with a matching color. Given Y, let $\mathcal{M}_Y \subseteq E$ denote the set of edges where Y makes a mistake. The objective is

$$\text{(MINECC)} \quad \min_Y \; \textstyle\sum_{e\in E} w_e \mathbb{1}_{\mathcal{M}_Y}(e), \tag{1}$$

where $\mathbb{1}_{\mathcal{M}_Y}$ is the indicator function for edge mistakes and the minimization is over all valid node colorings Y. The canonical LP relaxation for this objective is

$$\begin{array}{lll} \min & \sum_{e\in E} w_e x_e & \\ \text{s.t.} & \sum_{i=1}^{k} x_v^i = k-1 & \forall v \in V \\ & x_e \geq x_v^c & \text{if } e \in E_c \text{ where } c \in C \\ & 0 \leq x_v^i \leq 1 & \forall v \in V \text{ and } i \in C \\ & 0 \leq x_e \leq 1 & \forall e \in E. \end{array} \tag{2}$$

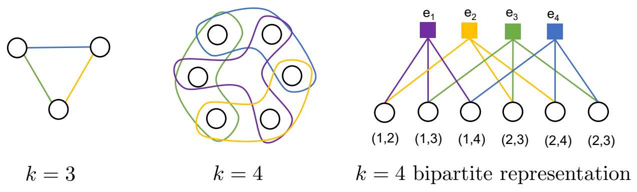

Figure 2. For integers $k \geq 3$ there is an edge-colored hypergraph for which the MINECC LP integrality gap is $2(1 - 1/k) = 2(1 - 1/(r+1))$. The instance involves k edges of different colors, and for each pair of distinct edges there is one node in the intersection. This can be visualized using a bipartite representation: there is a hyperedge-node (squares) for each color, and each standard node (circles) is defined by pairs of hyperedge-nodes.

The variable x_u^i can be interpreted as the distance between node u and color i. Every node color map Y can be translated into a binary feasible solution for this LP by setting $x_v^i = 0$ if $Y[v] = i$ and $x_v^i = 1$ otherwise, and by setting $x_e = 1$ when $e \in \mathcal{M}_Y$ and $x_e = 0$ otherwise. The constraints $x_e \geq x_v^c$ for $e \in E_c$ and the nonnegativity of w_e together imply that $x_e = \max_{v\in e} x_v^c$ at optimality.

2.1. Generic LP Rounding Algorithm

Algorithm 1 is our generic algorithm for rounding the MINECC LP relaxation, which takes an interval $I \subseteq [0, 1]$ as input. It generates a random threshold $\rho \in I$, and identifies the set of nodes v satisfying $x_v^i < \rho$ for each cluster i. If $x_v^i < \rho$, we say that color i "wants" node v and that i is a candidate color for node i. A node may have more than one candidate color, so we generate a random permutation of $\{1, 2, \ldots k\}$ that defines the priority of each color. In Algorithm 1, a color has higher priority if it comes later in the permutation π. We then define $Y[v]$ to be the color that has the highest priority among all nodes that want v. Nodes that are not wanted by any color are given an arbitrary color.

The rounding schemes of Amburg et al. (2020) amount to running Algorithm 1 with a fixed threshold (i.e., I is a single point). Here we use a random threshold from an interval I, similar to rounding strategies for convex relaxations of other multiway cut objectives (Călinescu et al., 2000; Ene et al., 2013; Chekuri & Madan, 2016). Our goal is to bound the probability that $e \in \mathcal{M}_Y$ for an arbitrary $e \in E$.

Observation 1. Let Y be the output of Algorithm 1 for some interval $I \subseteq [0, 1]$. If $\mathbb{P}[e \in \mathcal{M}_Y] \leq p x_e$ for every $e \in E$, the output node coloring Y is a p-approximate solution for MINECC, since the expected cost of Y is

$$\mathbb{E}\Big[\sum_{e\in E} w_e \mathbb{1}_{\mathcal{M}_Y}(e)\Big] = \sum_{e\in E} w_e \cdot \mathbb{P}[e \in \mathcal{M}_Y] \leq p \sum_{e\in E} w_e x_e.$$

Given this observation, in order to prove approximation guarantees for Algorithm 1, we will simply focus on bounding $\mathbb{P}[e \in \mathcal{M}_Y]$ under different conditions on I, r, and k.

Algorithm 1 GenColorRound(H, I)

```
    Input: $H = (V, E, C, \ell)$, interval $I \subseteq [0,1]$
    Output: Node coloring map $Y : V \to C$
    Solve the LP-relaxation (2)
    $\rho \leftarrow$ uniform random value in $I$
 5: $\pi \leftarrow$ uniform random permutation of $\{1, 2, \dots, k\}$
    For $i \in \{1, 2, \dots k\}$ define $S_i = \{v \in V : x_v^i < \rho\}$
    for $i = 1$ to $k$ do
      for $v \in S_{\pi(i)}$ do
        $Y[v] = \pi(i)$
10:   end for
    end for
    If $v \notin \bigcup_i S_i$, set $Y[v]$ to an arbitrary color
```

2.2. Linear Program Integrality Gap

Before proving new approximation guarantees, we establish a new integrality gap result for the canonical MinECC LP. This is related to integrality gap results for convex relaxations of Node-MC (Garg et al., 2004) and Hyper-MC (Ene et al., 2013). For these problems, an integrality gap was given only in terms of the number of clusters k. For MinECC, we establish a gap in terms of k and r simultaneously by considering a hypergraph where $k = r + 1$. See Figure 2 for an illustration and the appendix for a proof.

Lemma 2.1. *For every integer $k \geq 3$, there exists an instance $H = (V, E, C, \ell)$ of* MinECC *with $k = r + 1$ whose optimal solution makes $k - 1 = r$ mistakes, and for which the LP relaxation has a value of $\frac{k}{2} = \frac{r+1}{2}$. Thus, the integrality gap is $2\big(1 - \frac{1}{k}\big) = 2\big(1 - \frac{1}{r+1}\big)$.*

2.3. Comparison with the Node-MC LP Relaxation

Amburg et al. (2020) showed that MinECC reduces to Node-MC in an approximation-preserving way, implying a $2(1 - 1/k)$ approximation based on rounding the Node-MC LP relaxation. When $k \leq 2(r + 1)$, this is better than the approximation guarantee the authors obtain for directly rounding the canonical MinECC LP. Despite this, we show that the canonical LP is tighter than the Node-MC LP.

Theorem 2.2. *The value of the* MinECC *LP relaxation is always at least as large as (and can be strictly larger than) the lower bound obtained by reducing to* Node-MC *and using the* Node-MC *LP relaxation.*

Appendix D provides a more formal statement of this result and a proof, along with additional details on the relationship between MinECC and multiway cut objectives. In particular, we detail the approximation-preserving reductions from MinECC to Node-MC and Hyper-MC, and show that the best approximation results obtained via these reductions are in general not as strong as the approximation guarantees we develop using the MinECC LP relaxation.

3. Optimal LP Rounding for Hypergraphs

This section covers our optimal rounding scheme for the MinECC LP relaxation when $r > 2$, i.e., the hypergraph case. We provide our $\min\left\{2 - \frac{2}{k}, 2 - \frac{2}{r+1}\right\}$-approximation by combining two different approximation guarantees, obtained by running Algorithm 1 using two different intervals I. Most proofs are deferred to the appendix.

Proof overview. Throughout the section we consider an arbitrary fixed edge e with color $c = \ell(e)$ and LP variable $x_e = \max_{v \in e} x_v^c$. Recall that we say color i "wants" node v when $x_v^i < \rho$ for the randomly chosen threshold $\rho \in I$. In order to guarantee there is no mistake at e, color c must want every node in e, which is true if and only if $\rho > x_e$. Even if $\rho > x_e$, there is a chance of making a mistake at e if there are any other colors that want nodes from e. Our goal is to bound the probability of making a mistake at e in terms of x_e. If $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq p x_e$ for some p, then Observation 1 guarantees Algorithm 1 is a p-approximation.

The color threshold value. To aid in our results, we define the *first color threshold* of e to be $z_1^{(e)} = \min_{v \in e, i \neq c} x_v^i$. This is the smallest value such that $\rho > z_1^{(e)}$ implies there exists a color other than c that wants a node in e.

Observation 2. $1 - z_1^{(e)} \leq x_e$.

To prove the observation, let $j \neq c$ and $v \in e$ be chosen so that $z_1^{(e)} = x_v^j$. The first constraint in the MinECC LP implies that $\sum_{i=1}^{k} (1 - x_v^i) = 1 \implies 2 - x_v^j - x_v^c \leq 1$, so $1 \leq x_v^j + x_v^c \leq z_1^{(e)} + x_e$. We use this observation in the proofs of Theorems 3.1 and 3.2.

To simplify notation, we can write z_1 instead of $z_1^{(e)}$, while still noting that this value is specific to edge e. We keep the 1 in the subscript of z_1 since this is the *first* color threshold. We will define other color thresholds later for our rounding schemes for the graph case ($r = 2$), but this is not necessary for the results in this section.

Theorem 3.1. *Algorithm 1 with $I = (\frac{1}{2}, \frac{3}{4})$ is a $2\left(1 - \frac{1}{k}\right)$-approximation for* MinECC.

Proof. If $k = 2$, the constraint matrix for the LP relaxation is totally unimodular, so every basic feasible LP solution will have binary variables and our rounding procedure will find the optimal solution. Assume for the rest of the proof that $k \geq 3$. By Observation 1, it suffices to show that $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq 2\left(1 - \frac{1}{k}\right) x_e$. If $x_e \geq \frac{3}{4}$, then $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq 1 \leq \frac{4}{3} x_e \leq 2\left(1 - \frac{1}{k}\right) x_e$ for $k \geq 3$. We break up the remainder of the proof into two cases.

Case 1: $x_e < 1/2$. For $\rho \in (\frac{1}{2}, \frac{3}{4})$, color $c = \ell(e)$ will want all nodes in e because $x_e < \frac{1}{2}$. Observation 2 implies that $z_1 > \frac{1}{2}$. If $z_1 \geq \frac{3}{4}$, then no other color $i \neq c$ will want a node from e, meaning that $\mathbb{P}\left[e \in \mathcal{M}_Y\right] = 0$. If

$\frac{1}{2} < z_1 \leq \frac{3}{4}$, it is possible to make a mistake at e only if $\rho > z_1$, and even then the probability of making a mistake is at most $\frac{k-1}{k}$, since there is a $\frac{1}{k}$ probability that color c is given the highest priority by the random permutation π in Algorithm 1. Combining this with Observation 2 shows

$$\begin{aligned}\mathbb{P}\left[e \in \mathcal{M}_Y\right] &= \mathbb{P}\left[\rho > z_1\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho > z_1\right] \\ &\leq \frac{\frac{3}{4} - z_1}{\frac{3}{4} - \frac{1}{2}} \cdot \frac{k-1}{k} = 4\left(\frac{k-1}{k}\right)\left(1 - z_1 - \frac{1}{4}\right) \\ &\leq 4\left(1 - \frac{1}{k}\right)\left(x_e - \frac{x_e}{2}\right) = 2\left(1 - \frac{1}{k}\right)x_e.\end{aligned}$$

Case 2: $x_e \in \left[\frac{1}{2}, \frac{3}{4}\right)$**.** We apply a similar set of steps to see

$$\begin{aligned}\mathbb{P}\left[e \in \mathcal{M}_Y\right] &= \mathbb{P}\left[\rho \leq x_e\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho \leq x_e\right] \\ &\quad + \mathbb{P}\left[\rho > x_e\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho > x_e\right] \\ &= \frac{x_e - \frac{1}{2}}{\frac{3}{4} - \frac{1}{2}} \cdot 1 + \frac{\frac{3}{4} - x_e}{\frac{3}{4} - \frac{1}{2}} \cdot \frac{k-1}{k} \\ &= 4\left(\frac{x_e}{k} + \frac{k-3}{4k}\right) \leq 4\left(\frac{1}{k} + \frac{k-3}{2k}\right)x_e \\ &= 2\left(1 - \frac{1}{k}\right)x_e.\end{aligned}$$

□

Theorem 3.2 provides an approximation guarantee in terms of r, using a different interval. The main difference from Theorem 3.1 is that we use the fact that $\rho < \frac{2}{3}$ to bound the number of different colors that want a node in e, in terms of r. This allows us to bound the probability of making a mistake at e in terms of r instead of in terms of k.

Theorem 3.2. *Algorithm 1 with $I = (\frac{1}{2}, \frac{2}{3})$ is a $2(1 - \frac{1}{r+1})$-approximation for* MinECC *when $r > 2$.*

Combining Theorems 3.1 and 3.2 provides an optimal rounding strategy, matching the integrality gap from Lemma 2.1.

4. Optimal LP Rounding for Graphs

If $r = 2$, Theorem 3.2 can be slightly adjusted to prove that when $I = (\frac{1}{2}, \frac{2}{3})$, Algorithm 1 is a $\frac{3}{2}$-approximation for MinECC. Although this improves on the previous $\frac{5}{3}$-approximation (Amburg et al., 2020), it does not match the integrality gap of $\frac{4}{3}$ established by Lemma 2.1. This section shows how to use a larger interval $I = (\frac{1}{2}, \frac{7}{8})$ to obtain a $\frac{4}{3}$-approximation guarantee for the graph version ($r = 2$). The overall proof strategy mirrors our results for hypergraphs, but it requires several new technical results and a substantially more in-depth analysis.

4.1. Proof Setup, Overview, and Challenges

We consider a fixed arbitrary edge $e = (u, v)$ with color $c = \ell(e)$ and LP variable $x_e = \max\{x_u^c, x_v^c\}$. We will show that for every feasible solution to the LP relaxation,

$$\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq \frac{4}{3}x_e. \tag{3}$$

By Observation 1, this guarantees the $\frac{4}{3}$-approximation. Inequality (3) is easy to show for extreme values of x_e.

Lemma 4.1. *If $x_e \notin (\frac{1}{8}, \frac{3}{4})$, for Algorithm 1 with $I = (\frac{1}{2}, \frac{7}{8})$, then $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq \frac{4}{3}x_e$.*

Proof. If $x_e \geq \frac{3}{4}$, then $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq 1 \leq \frac{4}{3}x_e$. If $x_e \leq \frac{1}{8}$, then for every $\rho \in (\frac{1}{2}, \frac{7}{8})$, color $c = \ell(e)$ wants both u and v, and the constraint $\sum_{j=1}^{k} x_u^j = k-1$ can be used to show that no other color wants either u or v, so $\mathbb{P}\left[e \in \mathcal{M}_Y\right] = 0$. □

The following subsections cover the case $x_e \in (\frac{1}{8}, \frac{3}{4})$, which is far more challenging. The difficulty in proving inequality (3) for this case is that the probability of making a mistake depends heavily on the relationship among LP variables x_e and $\{x_u^i, x_v^i \colon i \in [k]\}$, and more specifically the relative ordering of these variables. Since k can be arbitrarily large, there can be many LP variables to consider, many possible orderings of these variables, and many different colors that want nodes u and v. The best strategy for bounding $\mathbb{P}\left[e \in \mathcal{M}_Y\right]$ depends on the configuration of LP variables, leading to an in-depth case analysis.

To provide a systematic proof for all cases, we first introduce a new set of variables $\{z_i\}_{i \in [k-1]}$ that can be viewed as a convenient rearrangement of the node-color distance variables $\{x_u^i, x_v^i \colon i \in [k]\}$. These generalize the *first color threshold* used in Section 3, and indicate the points at which there is a change in the number of distinct colors that "want" a node in e. We prove a key lemma on the relationship between x_e and the $\{z_i\}_{i \in [k-1]}$ variables (Lemma 4.2), and then show how to express $\mathbb{P}\left[e \in \mathcal{M}_Y\right]$ in terms of these variables for different possible feasible LP solutions. We finally prove $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq \frac{4}{3}x_e$ under different possible conditions by solving small auxiliary linear programs.

4.2. The Color Threshold Lemma

For $i \in \{0, 1, 2, \ldots, k-1\}$, the ith **color threshold** z_i is the smallest nonnegative value such that for every $\rho > z_i$, there are at least i distinct colors not equal to $c = \ell(e)$ that want a node in e. This definition does not require the i colors to all want the *same* node in e. More precisely, for each of the i colors, there exists a node $v \in e$ that is wanted by that color. By definition, the color threshold values are monotonic:

$$0 = z_0 \leq z_1 \leq z_2 \leq \cdots \leq z_{k-1} \leq 1. \tag{4}$$

These values make it easier to express the probability of making a mistake at e if we know $\rho > x_e$ and we know the value of ρ relative to the color threshold values. Formally,

$$\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho > x_e \text{ and } z_i < \rho \leq z_{i+1}\right] = \tfrac{i}{i+1}. \tag{5}$$

The following lemma is a generalization of Observation 1, though its proof is more involved.

Lemma 4.2. *For every integer $t \leq \frac{k}{2}$,*

$$t \leq x_e + z_t + z_{t+1} + \cdots + z_{2t-1}. \tag{6}$$

4.3. Auxiliary LPs for Bounding Probabilities

If we know where the color threshold values $\{z_i\}_{i \in [k-1]}$ are located relative to x_e and the endpoints of $I = (\frac{1}{2}, \frac{7}{8})$, we can derive an expression for $\mathbb{P}[e \in \mathcal{M}_Y]$ in terms of these values when applying Algorithm 1. The goal would then be to apply a sequence of carefully chosen inequalities to prove that this expression is bounded above by $\frac{4}{3}x_e$. This subsection shows how to accomplish this task when we already know the relative ordering of these variables. In the next subsection, we apply this strategy to guarantee that this holds for *all* possible orderings. The following lemmas both rely on Lemma 4.2.

Lemma 4.3. *If $x_e \in (\frac{1}{8}, \frac{1}{2})$ and $z_{p-1} \leq \frac{1}{2} \leq z_p \leq z_q \leq \frac{7}{8} \leq z_{q+1}$ for integers $p \leq q$, then $p = 1$, $q \leq 6$, and $\mathbb{P}[e \in \mathcal{M}_Y] \leq \frac{8}{3}A_q x_e$ where A_q is the optimal solution to*

$$\begin{array}{ll} \max & \dfrac{q}{q+1}\dfrac{7}{8}\chi - \displaystyle\sum_{j=1}^{q} \dfrac{1}{j(j+1)}\omega_j \\ \text{s.t.} & \textbf{\textit{(A\{i\})}}\ \omega_i - \omega_{i+1} \leq 0 \text{ for } i = 1, \dots, 5 \\ & \textbf{\textit{(A6)}}\ \chi - \omega_1 \leq 1 \\ & \textbf{\textit{(A7)}}\ 2\chi - \omega_2 - \omega_3 \leq 1 \\ & \textbf{\textit{(A8)}}\ 3\chi - 3\omega_5 \leq 1 \\ & \textbf{\textit{(A9)}}\ -\chi \leq -2 \quad \textbf{\textit{(A10)}}\ \omega_q - \frac{7}{8}\chi \leq 0. \end{array} \tag{7}$$

We derive an analogous result for the case $x_e \in [\frac{1}{2}, \frac{3}{4})$.

Lemma 4.4. *If $x_e \in [\frac{1}{2}, \frac{3}{4})$ and $z_{p-1} \leq x_e \leq z_p \leq z_q \leq \frac{7}{8} \leq z_{q+1}$ for integers $p \leq q$, then $p \leq 5$, $q \leq 10$, and $\mathbb{P}[e \in \mathcal{M}_Y] \leq \frac{8}{3}B_{p,q}x_e$ where $B_{p,q}$ is the solution to*

$$\begin{array}{ll} \max & \dfrac{1}{p} + \left(\dfrac{q}{q+1}\dfrac{7}{8} - \dfrac{1}{2}\right)\chi - \displaystyle\sum_{j=p}^{q} \dfrac{1}{j(j+1)}\omega_j \\ \text{s.t.} & \textbf{\textit{(B\{i\})}}\ \omega_i - \omega_{i+1} \leq 0 \text{ for } i = 1, \dots, 9 \\ & \textbf{\textit{(B10)}}\ \chi - \omega_1 \leq 1 \\ & \textbf{\textit{(B11)}}\ 2\chi - \omega_2 - \omega_3 \leq 1 \\ & \textbf{\textit{(B12)}}\ 3\chi - \omega_3 - \omega_4 - \omega_5 \leq 1 \\ & \textbf{\textit{(B13)}}\ 4\chi - 4\omega_7 \leq 1 \quad \textbf{\textit{(B14)}}\ \omega_{p-1} \leq 1 \\ & \textbf{\textit{(B15)}}\ -\omega_p \leq -1 \quad \textbf{\textit{(B16)}}\ \omega_q - \frac{7}{8}\chi \leq 0. \end{array} \tag{8}$$

In order to prove that $\mathbb{P}[e \in \mathcal{M}_Y] \leq \frac{4}{3}x_e$, it remains to show that for all valid choices of p and q, the optimal solutions LP (7) and LP (8) are bounded above by $\frac{1}{2}$. The result will then follow by Lemmas 4.3 and 4.4. The difficulty is that there are many valid choices of p and q to check, each of which requires a different linear program. When $x_e \in (\frac{1}{8}, \frac{1}{2})$, Lemma 4.3 indicates that $p = 1$ and $q \in \{1, 2, 3, 4, 5, 6\}$, so we must consider six cases. When $x_e \in [\frac{1}{2}, \frac{3}{4})$, Lemma 4.4 shows that any pair (p, q) satisfying $p \leq 5$, $q \leq 10$, and $p \leq q$ is a valid case, and there are 40 such pairs. If we are content with simply computing numerical solutions for each case, we can quickly confirm that the optimal solution for each of the 46 linear programs is bounded above by $\frac{1}{2}$. In the next subsection, we will show how to obtain a complete *analytical* proof by using LP duality theory to extract a set of inequalities for each case that will prove an upper bound on the solution to each auxiliary LP. Note that there are many additional valid constraints that we could add to LPs (7) and (8). We have omitted some constraints and have simplified others in order to obtain the smallest and simplest constraint set that suffices to prove $\mathbb{P}[e \in \mathcal{M}_Y] \leq \frac{4}{3}x_e$. Adding more constraints makes the analysis more cumbersome without improving the bound.

4.4. Using LP Duality for Case Analysis

For each valid choice of q, we can use LP duality to bound the optimal solution of LP (7) above by $\frac{1}{2}$. The same basic steps also work for LP (8). The dual of LP (7) is another LP with a variable for each constraint **A{i}**. By LP duality theory, every feasible solution to the dual provides an upper bound on the primal LP objective, so it suffices to find a set of dual variables with a dual LP value of $\frac{1}{2}$ or less.

Proof for $q = 1$. We illustrate this for LP (7) when $q = 1$. The primal LP objective for this case is $\frac{7}{16}\chi - \frac{1}{2}\omega_1$, and we must bound this above by $\frac{1}{2}$. Computing an optimal solution for the dual LP, we find that the constraint **(A6)** corresponds to a dual variable of $\frac{1}{2}$, constraint **(A9)** has a dual variable of $\frac{1}{16}$, and all other dual variables are zero. Multiplying constraints by the dual variables and summing produces a sequence of inequalities that proves the desired result:

$$\textbf{(A6)}\ \chi - \omega_1 \leq 1 \text{ and } \textbf{(A9)}\ -\chi \leq -2 :$$
$$\implies \frac{1}{2}(\chi - \omega_1) + \frac{1}{16}(-\chi) \leq \frac{1}{2}(1) + \frac{1}{16}(-2)$$
$$\implies \frac{7}{16}\chi - \frac{1}{2}\omega_1 \leq \frac{3}{8} \leq \frac{1}{2} \text{ for all valid } \chi, \omega_1.$$

Table 3 in Appendix C provides dual variables for LP (7) for all other choices of q, and outlines a similar strategy for bounding LP (8). We conclude the following result.

Lemma 4.5. *For any integer $q \in \{1, 2, \dots, 6\}$, the optimal solution to linear program (7) is at most $\frac{1}{2}$, and for any pair of integers (p, q) satisfying $1 \leq p \leq 5$ and $p \leq q \leq 10$, the optimal solution to linear program (8) is at most $\frac{1}{2}$. Hence, for every $x_e \in (\frac{1}{8}, \frac{3}{4})$, $\mathbb{P}[e \in \mathcal{M}_Y] \leq \frac{4}{3}x_e$ where Y is the node coloring returned by Algorithm 1 when $I = (\frac{1}{2}, \frac{7}{8})$.*

Our main approximation result for $r = 2$ is now just a corollary of Lemmas 4.1 and 4.5 and Observation 1.

Theorem 4.6. *Algorithm 1 with $I = (\frac{1}{2}, \frac{7}{8})$ is a $\frac{4}{3}$-approximation algorithm for* COLOR-EC.

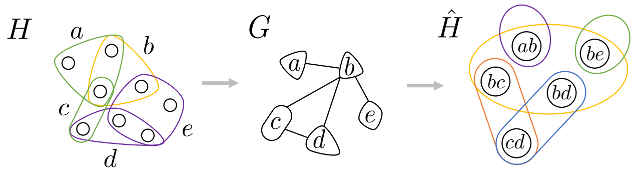

Figure 3. MinECC can be reduced to Vertex Cover by replacing each hyperedge with a node and adding an edge between nodes that represent overlapping hyperedges of different colors. Vertex Cover on a graph G can be reduced to MinECC by introducing a node for each edge in G, and a hyperedge of a unique color for each node-neighborhood in G.

The appendix details several challenges that would need to be overcome in order to avoid a lengthy case analysis when trying to prove a $\frac{4}{3}$-approximation. Among other challenges, we prove the following result, indicating that if we apply Algorithm 1 with a different interval I, it would either become more challenging or impossible (depending on the interval I) to prove inequality (3).

Lemma 4.7. *Let Y denote the node color map returned by running Algorithm 1 with $I = (a, b) \subseteq (0, 1)$. If $a > \frac{1}{2}$, there exists a feasible LP solution such that $\mathbb{P}[e \in \mathcal{M}_Y] > \frac{4}{3}x_e$. If $a \leq \frac{1}{2}$ and $b < \frac{7}{8}$, there exists a feasible LP solution such that $\mathbb{P}[e \in \mathcal{M}_Y] > \frac{4}{3}x_e$.*

5. Vertex Cover Equivalence and Algorithms

Cai & Leung (2018) showed that when $r = 2$, MinECC can be reduced in an approximation-preserving way to Vertex Cover. We not only extend this result to $r > 2$, but also prove an approximation-preserving reduction in the other direction that only applies if we consider the hypergraph ($r > 2$) setting. This immediately leads to new combinatorial approximation algorithms and refined hardness results. Explicitly forming the reduced Vertex Cover instance and applying existing Vertex Cover algorithms leads to runtimes that scale superlinearly in terms of $\sum_{e \in E} |e|$ (the size of the hypergraph). As a key algorithmic contribution, we design careful implicit implementations of existing Vertex Cover algorithms to provide 2-approximations for MinECC that have a runtime of $O(\sum_{e \in E} |e|)$.

5.1. Vertex Cover Equivalence

The equivalence between *hypergraph* MinECC and Vertex Cover is summarized in two theorems (see Figure 3).

Theorem 5.1. *Let $G = (V, E)$ be a graph with maximum degree Δ.* Vertex Cover *on G can be reduced in an approximation-preserving way to* MinECC *on an edge-colored hypergraph H with rank $r = \Delta$.*

Theorem 5.2. *An instance $H = (V, E, C, \ell)$ of* MinECC *can be reduced in an approximation-preserving way to an instance of* Vertex Cover *on a graph G with $|E|$ nodes.*

Theorem 5.1 implies new hardness results: assuming r and k are arbitrarily large, it is NP-hard to approximate hypergraph MinECC to within a factor better than 1.3606 (Dinur & Safra, 2005), and UGC-hard to approximate to within a factor that is a constant amount smaller than 2 (Khot & Regev, 2008). The relationship between maximum degree in G and hypergraph rank in H implies furthermore that it is UGC-hard to obtain a $2 - (2 + o_r(1))\frac{\ln \ln r}{\ln r}$ approximation for rank-r MinECC (assuming arbitrarily large k), using the Vertex Cover hardness results of Austrin et al. (2011). This result also implies that the maximum independent set problem is reducible in an approximation-preserving way to hypergraph MaxECC, indicating that it is NP-hard to obtain an approximation for hypergraph MaxECC that is sublinear in terms of the number of edges in the hypergraph (Zuckerman, 2006).

Theorem 5.2 generalizes the reduction from *graph* MinECC to Vertex Cover shown by Cai & Leung (2018). This can be seen by viewing MinECC as an edge deletion problem. In an edge-colored hypergraph $H = (V, E, C, \ell)$, we say that $(e, f) \in E \times E$ is a *bad edge pair* if e and f overlap and have different colors. The MinECC objective is then equivalent to deleting a minimum weight set of edges so that no bad edge pairs remain. A simple 2-approximation for MinECC can be designed by explicitly converting the edge-colored hypergraph into a graph and applying existing combinatorial Vertex Cover algorithms. Visiting each node $v \in V$ and then iterating through all pairs of hyperedges incident to v in H takes $O(\sum_{v \in V} d_v^2)$ time where d_v is the degree of node v. The fastest algorithms for (unweighted *and* weighted) Vertex Cover take linear-time in terms of the number of edges in the graph (Pitt, 1985; Bar-Yehuda & Even, 1985). There can be up to $O(|E|^2)$ edges in the reduced graph G, so this approach has a runtime of $O(|E|^2 + \sum_{v \in V} d_v^2)$.

5.2. Linear-time 2-Approximation Algorithm

It would be ideal to develop an algorithm for MinECC whose runtime is not just linear in terms of the number of edges in a reduced graph, but is linear *in terms of the hypergraph size*, i.e., $O(\sum_{e \in E} |e|)$. This may seem inherently challenging, given that existing Vertex Cover algorithms rely on explicitly visiting all edges in a graph. However, this can be accomplished using a careful *implicit* implementation of a Vertex Cover algorithm. Pitt's algorithm (Pitt, 1985) is a simple randomized 2-approximation for weighted Vertex Cover that iteratively visits edges in a graph $G = (V_G, E_G)$ in an arbitrary order. Each time it encounters an uncovered edge, it samples one of the two endpoints (in proportion to the node weight) to include in the cover. We will design a variant of Pitt's algorithm that we can apply directly to a MinECC instance H without ever forming G.

Additional notation. Assume a fixed ordering of edges $E = \{e_1, e_2, \ldots, e_{|E|}\}$. For notational simplicity, let $w(i) = w_{e_i}$ and $\ell(i) = \ell(e_i)$ denote the weight and color of the ith edge, respectively. For each node $v \in V$, using a slight abuse of notation let $v(j)$ denote the index of the jth edge that is adjacent to v. These edge indices will be stored in a list $L_E(v) = [v(1),\ v(2),\ \cdots,\ v(d_v)]$ where d_v is the degree of v. For example, if node v is in three hyperedges $\{e_2, e_5, e_9\}$, then $L_E(v) = [2,\ 5,\ 9]$. We assume that edges are ordered by color, so that in $L_E(v)$, all of the indices for edges of color 1 come first, then edges with color 2, etc (see Figure 4). If the hypergraph is not stored this way, it can easily be re-arranged in $O(\sum_{e \in E} |e|)$ time to satisfy this property. Our main goal is to obtain a set $\mathcal{D}$ of edge indices to delete that implicitly corresponds to an approximate vertex cover in G. Once we have obtained such an edge set $\mathcal{D}$, we can iterate through all edges in H, and for each $e \in E - \mathcal{D}$ we can assign all nodes in e to have color $\ell(e)$, which takes $O(\sum_{e \in E} |e|)$ time. It remains to show how we can efficiently obtain a set $\mathcal{D}$ that implicitly encodes a 2-approximate vertex cover in G.

Implicit implementation. Iterating through edges in G is equivalent to iterating through bad edge pairs in H. A bad edge pair is "covered" if one of the edges is added to $\mathcal{D}$. Pitt's algorithm visits edges in G in an arbitrary order, so we can traverse bad edge pairs in H in any way that is convenient. We choose to iterate through nodes, and for each node $v \in V$ we will consider all bad edge pairs that contain v in their intersection. Explicitly visiting all pairs of edges incident to v takes $O(d_v^2)$ time. However, deleting an edge will cover multiple bad edge pairs at once, so we will be able to "skip" many pairs to speed up the process. To accomplish this, we maintain pointers to the front (f) and back (b) of $L_E(v) = [v(1),\ v(2),\ \cdots,\ v(d_v)]$ which we initialize to $f = 1$ and $b = d_v$. Unless v is only adjacent to nodes of one color, in the first step we know $(e_{v(f)}, e_{v(b)})$ will be a bad edge pair that needs to be covered. Using Pitt's technique, we randomly sample one edge to delete based on its weight. If we delete $e_{v(f)}$, then we no longer need to consider any other bad edge pairs involving $e_{v(f)}$, so we update $f \leftarrow f + 1$ and consider the next edge in the list. Otherwise, we delete edge $e_{v(b)}$ and decrement the back pointer: $b \leftarrow b - 1$. If at any point we encounter an edge that was added to $\mathcal{D}$ previously, we move past it by incrementing f or decrementing b. We stop this procedure when $\ell(v(f)) = \ell(v(b))$. Even if $f \neq b$, our assumption that edges are ordered by color means that there are no more bad edge pairs containing v to consider. Figure 4 is an illustration of this process. Since we update either f or b in each step, and each step involves $O(1)$ operations, covering all bad edge pairs containing v takes $O(d_v)$ time. We refer to this algorithm as PittColoring (see Appendix E for pseudocode), and end with a summarizing theorem.

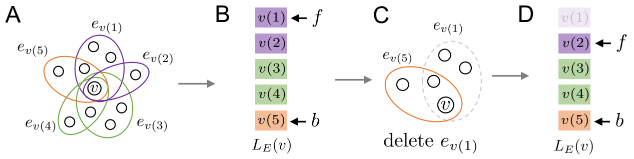

Figure 4. An illustration for covering bad edge pairs containing a node v. (A) v is contained in 5 edges and 8 bad edge pairs. (B) Consider edges at opposite ends of v's edge list $L_E(v)$, ordered by color. Edges $e_{v(1)}$ and $e_{v(5)}$ define a bad edge pair. (C) Given a bad edge pair, sample one to delete. (D) Once one $e_{v(1)}$ is deleted, we no longer consider bad edge pairs containing this edge, so we can advance the pointer f to the next edge in $L_E(v)$.

Table 1. Approximation factors (ratio between algorithm output and LP lower bound) and runtimes obtained on five benchmark edge-colored hypergraphs. MajorityVote (MV) is deterministic. Our vertex cover algorithms PittColoring and MatchColoring both involve some amount of randomization, so we list the mean values and standard deviations over 50 runs.

	Ratio to LP lower bound			
Dataset	LP	MV	PittColoring	MatchColoring
`Brain`	1.0	1.01	1.07 ±0.01	1.08 ±0.01
`Cooking`	1.0	1.21	1.23 ±0.01	1.23 ±0.0
`DAWN`	1.0	1.09	1.57 ±0.04	1.58 ±0.03
`MAG-10`	1.0	1.18	1.39 ±0.01	1.49 ±0.0
`Walmart`	1.0	1.2	1.13 ±0.0	1.18 ±0.0
	Runtime (in seconds)			
Dataset	LP	MV	PittColoring	MatchColoring
`Brain`	0.52	0.001	0.006 ±0.028	0.002 ±0.001
`Cooking`	127	0.002	0.01 ±0.003	0.008 ±0.007
`DAWN`	4.23	0.003	0.01 ±0.004	0.005 ±0.003
`MAG-10`	17.0	0.012	0.04 ±0.006	0.04 ±0.016
`Walmart`	321	0.07	0.053 ±0.007	0.05 ±0.009

Theorem 5.3. *PittColoring is a randomized 2-approximation algorithm for weighted* MINECC *that runs in $O(\sum_{v \in V} d_v) = O(\sum_{e \in E} |e|)$ time.*

Our strategy for iterating through bad edge pairs can also be used in conjunction with other VERTEX COVER algorithms. If edges are unweighted, every time we visit an uncovered bad edge pair in the list $L_E(v)$ we can instead delete *both* edges and update both pointers f and b. This leads to a new 2-approximation algorithm MatchColoring for the unweighted objective (see Appendix E).

6. Implementations and Experiments

Although our primary focus is to improve the theoretical foundations of edge-colored clustering, our algorithms are also easy to implement and very practical.

Results on benchmark datasets. Table 1 reports algorithmic results on the benchmark edge-colored hypergraphs

Table 2. Results (averaged over 50 runs) on the `Trivago` hypergraph. For PittColoring, the *Apx* column gives the theoretical expected approximation guarantee. For other algorithms, *Apx* is an a posteriori guarantee, improving on the method's theoretical guarantee, obtained using a lower bound computed by the algorithm.

Algorithm	*Mistakes*	*Sat*	*Apx*	*Acc*	*Run*
PittColoring	74624	0.70	2	0.72	0.13
MatchColoring	77606	0.69	1.74	0.70	0.12
MajorityVote	67941	0.73	36.43	0.77	0.11
Hybrid	65192	0.74	1.46	0.78	0.23
LP	58031	0.77	1.001	0.80	1549

of Amburg et al. (2020). On all datasets but `Walmart`, the MINECC LP relaxation produces integral solutions, i.e., we find the optimal MINECC solution without rounding. Even for `Walmart`, the LP solution is nearly integral and a very simple rounding scheme (namely, for each $v \in V$, assign it color $i^* = \text{argmin}_i \, x_v^i$), produces a solution that is within a factor 1.00003 of optimal. For this reason, while our new LP rounding techniques improve the best theoretical results, is it not meaningful to compare them empirically against previous rounding techniques. It is however very meaningful to compare our vertex cover based algorithms against alternative approaches. Table 1 shows that these algorithms are orders of magnitude faster than solving and rounding the LP relaxation, and still obtain good approximate solutions. They also have the same asymptotic runtime as MajorityVote, a fast previous combinatorial algorithm that comes with a much worse r-approximation guarantee. Statistics for these datasets, additional implementation details, and more experimental results are provided in the appendix.

Results for new `Trivago` hypergraph. Our empirical contributions include a new benchmark dataset for edge-colored clustering, and a very practical hybrid algorithm that combines the strengths of MatchColoring and MajorityVote. The dataset is a hypergraph with 207974 nodes, 247362 edges, rank $r = 85$, and $k = 55$ colors. Nodes correspond to vacation rentals on the booking website `Trivago`, and each edge corresponds to the set of rentals that a user clicks on during a single user browsing session. Edge colors correspond to countries where the browsing session happens (e.g., *Trivago.com* is the USA platform and *Trivago.es* is for Spain). Nodes also come with ground truth colors that define the country the vacation rental is located in. Because users often (even if not always) search for vacation rentals within their own country, edge colors and ground truth node colors tend to match. This therefore serves as a useful new benchmark dataset for the ECC objective, that is much larger than previous benchmarks.

Table 2 shows results for each algorithm on the `Trivago` hypergraph, including the number of mistakes made, the proportion of satisfied edges (*Sat*), the method's approximation guarantee (*Apx*), the number of nodes labeled correctly with respect to the ground truth (*Acc*, i.e., accuracy), and the runtime (*Run*, in seconds). PittColoring, MatchColoring and MajorityVote produce good results very quickly. A new algorithm Hybrid obtains even better results by first applying the edge deletion step of MatchColoring and then using the MajorityVote assignment for nodes that are isolated after edge deletions. MajorityVote is only guaranteed to return an 85-approximation since $r = 85$, while MatchColoring and Hybrid have a deterministic 2-approximation guarantee. We obtain improved a posteriori approximation guarantees for these three algorithms by comparing the number of mistakes they make against a lower bound on the optimal solution that can be computed based on the output of each algorithm. PittColoring has an *expected* 2-approximation guarantee, but does not produce an explicit lower bound so it does not come with improved a posteriori guarantees.

Solving and rounding the LP relaxation produces a very good output on the `Trivago` dataset, with an a posteriori approximation guarantee of 1.0014, an edge satisfaction of 0.765, and an accuracy of 0.80. However, it takes around 25 minutes to solve the LP relaxation, which is roughly four orders of magnitude slower than our linear-time combinatorial algorithms. This result does indicate that LP-based methods for MINECC produce great results when it is possible to run them. However, our combinatorial algorithms will be able to scale to extremely large datasets where it is infeasible to rely on LP-based techniques. More details for all of our experimental results are provided in Appendices E and F.

7. Conclusion and Discussion

We have presented improved algorithms and hardness results for Edge-Colored Clustering. Our combinatorial algorithms have asymptotically optimal runtimes, and if k and r are arbitrarily large, their 2-approximation guarantee is the best possible assuming the unique games conjecture. Our LP algorithms are also tight with respect to the integrality gap. One open question is to determine whether alternative techniques could lead to improved approximations for fixed values of r or k. Previous work has shown that the optimal approximation factor for NODE-MC and HYPER-MC coincides with the integrality gap of their LP relaxation, assuming the unique games conjecture (Ene et al., 2013). An open question is whether a similar result holds for the MINECC LP. In other words, is it UGC-hard to approximate MINECC below the LP integrality gap? We also proved that the MINECC LP relaxation is tighter than the NODE-MC relaxation. An open question in this direction is to better understand the relationship between the MINECC LP and relaxations obtained by considering more general problems, such as the Lovász relaxation for submodular multiway partition (Chekuri & Ene, 2011b) or the basic LP for minimum constraint satisfaction problems (Ene et al., 2013).

References

Ageev, A. and Kononov, A. Improved approximations for the max k-colored clustering problem. In *International Workshop on Approximation and Online Algorithms*, pp. 1–10. Springer, 2014.

Ageev, A. and Kononov, A. A 0.3622-approximation algorithm for the maximum k-edge-colored clustering problem. In *International Conference on Mathematical Optimization Theory and Operations Research*, pp. 3–15. Springer, 2020.

Alhamdan, Y. M. and Kononov, A. Approximability and inapproximability for maximum k-edge-colored clustering problem. In *International Computer Science Symposium in Russia*, pp. 1–12. Springer, 2019.

Amburg, I., Veldt, N., and Benson, A. Clustering in graphs and hypergraphs with categorical edge labels. In *Proceedings of The Web Conference 2020*, pp. 706–717, 2020.

Amburg, I., Veldt, N., and Benson, A. R. Diverse and experienced group discovery via hypergraph clustering. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, SDM '22, pp. 145–153, 2022. doi: 10.1137/1.9781611977172.17. URL `https://epubs.siam.org/doi/abs/10.1137/1.9781611977172.17`.

Anava, Y., Avigdor-Elgrabli, N., and Gamzu, I. Improved theoretical and practical guarantees for chromatic correlation clustering. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pp. 55–65, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-3469-3. doi: 10.1145/2736277.2741629. URL `https://doi.org/10.1145/2736277.2741629`.

Angel, E., Bampis, E., Kononov, A., Paparas, D., Pountourakis, E., and Zissimopoulos, V. Clustering on k-edge-colored graphs. *Discrete Applied Mathematics*, 211: 15–22, 2016.

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

Bansal, N., Blum, A., and Chawla, S. Correlation clustering. *Machine Learning*, 56:89–113, 2004.

Bar-Yehuda, R. and Even, S. A local-ratio theorm for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.

Bonchi, F., Gionis, A., Gullo, F., and Ukkonen, A. Chromatic correlation clustering. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pp. 1321–1329, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1462-6. doi: 10.1145/2339530.2339735. URL `http://doi.acm.org.ezproxy.lib.purdue.edu/10.1145/2339530.2339735`.

Bonchi, F., Gionis, A., Gullo, F., Tsourakakis, C. E., and Ukkonen, A. Chromatic correlation clustering. *ACM Trans. Knowl. Discov. Data*, 9(4):34:1–34:24, June 2015. ISSN 1556-4681. doi: 10.1145/2728170. URL `http://doi.acm.org.ezproxy.lib.purdue.edu/10.1145/2728170`.

Cai, L. and Leung, O. Y. Alternating path and coloured clustering. *arXiv preprint arXiv:1807.10531*, 2018.

Chekuri, C. and Ene, A. Approximation algorithms for submodular multiway partition. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pp. 807–816. IEEE, 2011a.

Chekuri, C. and Ene, A. Submodular cost allocation problem and applications. In *International Colloquium on Automata, Languages, and Programming*, pp. 354–366. Springer, 2011b.

Chekuri, C. and Madan, V. Simple and fast rounding algorithms for directed and node-weighted multiway cut. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 797–807. SIAM, 2016.

Crossley, N. A., Mechelli, A., Vértes, P. E., Winton-Brown, T. T., Patel, A. X., Ginestet, C. E., McGuire, P., and Bullmore, E. T. Cognitive relevance of the community structure of the human brain functional coactivation network. *Proceedings of the National Academy of Sciences*, 110(28):11583–11588, 2013. ISSN 0027-8424. doi: 10.1073/pnas.1220826110. URL `https://www.pnas.org/content/110/28/11583`.

Călinescu, G., Karloff, H., and Rabani, Y. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564 – 574, 2000. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1999.1687. URL `http://www.sciencedirect.com/science/article/pii/S0022000099916872`.

Dahlhaus, E., Johnson, D. S., Papadimitriou, C. H., Seymour, P. D., and Yannakakis, M. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23: 864–894, 1994.

Dinur, I. and Safra, S. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pp. 439–485, 2005.

Ene, A., Vondrák, J., and Wu, Y. Local distribution and the symmetry gap: Approximability of multiway partitioning problems. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 306–325. SIAM, 2013.

Fountoulakis, K., Li, P., and Yang, S. Local hyper-flow diffusion. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 27683–27694. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/e924517087669cf201ea91bd737a4ff4-Paper.pdf.

Garg, N., Vazirani, V. V., and Yannakakis, M. Multiway cuts in node weighted graphs. *Journal of Algorithms*, 50 (1):49–61, 2004.

Hein, M., Setzer, S., Jost, L., and Rangapuram, S. S. The total variation on hypergraphs - learning on hypergraphs revisited. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 2427–2435, USA, 2013. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=2999792.2999883.

Kellerhals, L., Koana, T., Kunz, P., and Niedermeier, R. Parameterized algorithms for colored clustering. In *Proceedings of The 2023 AAAI Conference on Artificial Intelligence*, 2023.

Khot, S. and Regev, O. Vertex cover might be hard to approximate to within 2- ε. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

Klodt, N., Seifert, L., Zahn, A., Casel, K., Issac, D., and Friedrich, T. A color-blind 3-approximation for chromatic correlation clustering and improved heuristics. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 882–891, 2021.

Li, P. and Milenkovic, O. Inhomogeneous hypergraph clustering with applications. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2308–2318. Curran Associates, Inc., 2017.

Li, P. and Milenkovic, O. Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering. In *International Conference on Machine Learning*, pp. 3014–3023. PMLR, 2018.

Pitt, L. B. *A simple probabilistic approximation algorithm for vertex cover*. Yale University, Department of Computer Science, 1985.

Xiu, Q., Han, K., Tang, J., Cui, S., and Huang, H. Chromatic correlation clustering, revisited. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=jjJgLNrCQB.

Zuckerman, D. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pp. 681–690, 2006.

A. Extended Related Work

Chromatic correlation clustering. The graph version of ECC is closely related to chromatic correlation clustering (Bonchi et al., 2015; Anava et al., 2015; Klodt et al., 2021; Xiu et al., 2022), an edge-colored generalization of correlation clustering (Bansal et al., 2004). Chromatic correlation clustering also takes an edge-colored graph as input and seeks to cluster nodes in a way that minimizes edge mistakes. ECC and chromatic correlation clustering both include penalties for (1) separating two nodes that share an edge, and (2) placing an edge of one color in a cluster of a different color. The key difference is that chromatic correlation clustering also includes a penalty for placing two non-adjacent nodes in the same cluster. In other words, chromatic correlation clustering interprets a non-edge as an indication that two nodes are dissimilar and should not be clustered together, whereas MINECC treats non-edges simply as missing information and does not include this type of penalty. As a result, a solution to chromatic correlation clustering may involve multiple different clusters of the same color, rather than one cluster per color. Unlike MINECC, chromatic correlation clustering is known to be NP-hard even in the case of a single color. Various constant-factor approximation algorithms have been designed, culminating in a 2.5-approximation (Xiu et al., 2022), but these do not apply to the general weighted case. Another difference is that there are no results for chromatic correlation clustering in hypergraphs.

Multiway cut and partition problems. The reduction from MINECC to a special case of NODE-MC situates MINECC within a broad class of multiway cut and multiway partition problems (Ene et al., 2013; Garg et al., 2004; Dahlhaus et al., 1994; Călinescu et al., 2000; Chekuri & Ene, 2011a;b; Chekuri & Madan, 2016). For NODE-MC (Garg et al., 2004), one is given a node-weighted graph $G = (V, E)$ with k special terminal nodes, and the goal is to remove a minimum weight set of nodes to disconnect all terminals. NODE-MC is approximation equivalent to the hypergraph multiway cut problem (HYPER-MC) (Chekuri & Ene, 2011b), where one is give a hypergraph $H = (V, E)$ and k terminal nodes, and the goal is to remove the minimum weight set of edges to separate terminals. These problems generalize the standard edge-weighted multiway cut problem in graphs (GRAPH-MC) (Dahlhaus et al., 1994), where the goal is to separate k terminal nodes in a graph by removing a minimum weight set of edges. NODE-MC and HYPER-MC are also special cases of the more general submodular multiway partition problem (SUBMODULAR-MP), which has a best known approximation factor of $2(1 - 1/k)$ (Ene et al., 2013). This matches the best approximation guarantee for NODE-MC but requires solving and rounding a generalized Lovász relaxation (Chekuri & Ene, 2011b). It is worth noting that when $k = 2$, MINECC can be reduced to the minimum s-t cut problem, i.e., the 2-terminal version of GRAPH-MC, but this does not generalize to $k > 2$. Amburg et al. (2020) showed a way to approximate MINECC by approximating a related instance of GRAPH-MC, but the objectives differ by a factor $(r + 1)/2$.

B. Proofs for Linear Programming Results

Proof of Lemma 2.1 (LP integrality gap)

Proof. We will construct a hypergraph H that has exactly one edge e_c for each color $c \in \{1, 2, \ldots, k\}$, and a total of $\binom{k}{2}$ nodes, each of which participates in exactly two edges. More precisely, we index each node by a pair of color indices $(i, j) \in C \times C$ with $i \neq j$. We place node (i, j) in the hyperedge e_i and the hyperedge e_j. Each edge contains exactly $r = k - 1$ nodes. The optimal ECC solution makes a mistake at all but one edge. To see why, if we do not make a mistake at e_c, this means every node in e_c is given label c. For each color $i \neq c$, the node with index (i, c) is in e_c and e_i, which means we will make a mistake at e_i since this node was given label c. Meanwhile, the optimal solution to the LP relaxation is $\frac{k}{2}$: for node v corresponding to color pair (i, j), we set $x_v^i = x_v^j = \frac{1}{2}$ and $x_v^c = 1$ for every $c \notin \{i, j\}$. Thus, $x_e = \frac{1}{2}$ for each hyperedge $e \in E$, for an overall LP objective of $\frac{k}{2}$. □

Proof of Theorem 3.2 (Hypergraph approximation result)

Proof. Let e be an arbitrary edge with color $c = \ell(e)$. We need to show that $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq 2\left(1 - \frac{1}{r+1}\right)x_e$. If $x_e \geq \frac{2}{3}$, this is trivial since $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq 1 \leq \frac{3}{2}x_e \leq 2\left(1 - \frac{1}{r+1}\right)x_e$ as long as $r \geq 3$. We cover two remaining cases.

Case 1: $x_e < \frac{1}{2}$**.** For every $\rho \in (\frac{1}{2}, \frac{2}{3})$, color c wants all nodes in e, and Observation 2 implies that $z_1 > 1/2$. Therefore, the algorithm can only make a mistake at e if ρ falls between z_1 and $\frac{2}{3}$. This never happens if $z_1 \geq \frac{2}{3}$, so assume that $z_1 < \frac{2}{3}$. Because $\rho < \frac{2}{3}$, for every $v \in e$ it is possible for two different colors to want v, but never three colors. Since color c is guaranteed to want every $v \in e$, the worse case scenario is when $\rho > z_1$ and each $v \in e$ is wanted by a different color that is unique to that node. In this case, the probability of making a mistake at e is at most $\frac{r}{r+1}$. Putting all of the pieces together we have

$$\begin{aligned}\mathbb{P}\left[e \in \mathcal{M}_Y\right] &= \mathbb{P}\left[\rho > z_1\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho > z_1\right] \\ &= \frac{\frac{2}{3} - z_1}{\frac{2}{3} - \frac{1}{2}} \cdot \frac{r}{r+1} = \frac{6r}{r+1}\left(1 - z_1 - \frac{1}{3}\right) \\ &\leq \frac{6r}{r+1}\left(x_e - \frac{2}{3}x_e\right) = 2\left(1 - \frac{1}{r+1}\right)x_e.\end{aligned}$$

Case 2: $x_e \in \left[\frac{1}{2}, \frac{2}{3}\right)$**.** If $\rho \leq x_e$, we are guaranteed to make a mistake at e, but this happens with bounded probability. If $\rho > x_e$, we can argue as in Case 1 that the probability of making a mistake at e is at most $\frac{r}{r+1}$. We therefore have:

$$\begin{aligned}\mathbb{P}\left[e \in M\right] &= \mathbb{P}\left[\rho \leq x_e\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho \leq x_e\right] \\ &\quad + \mathbb{P}\left[\rho > x_e\right]\mathbb{P}\left[e \in \mathcal{M}_Y \mid \rho > x_e\right] \\ &= \frac{x_e - \frac{1}{2}}{\frac{2}{3} - \frac{1}{2}} \cdot 1 + \frac{\frac{2}{3} - x_e}{\frac{2}{3} - \frac{1}{2}} \cdot \frac{r}{r+1} \\ &= 6\left(x_e - \frac{1}{2} + \frac{2r}{3r+3} - \frac{x_e r}{r+1}\right) \\ &= 6\left(\frac{x_e}{r+1} + \frac{r-3}{6r+6}\right) \\ &\leq 6\left(\frac{3x_e}{3r+3} + \frac{(r-3)x_e}{3r+3}\right) = 2\left(1 - \frac{1}{r+1}\right)x_e.\end{aligned}$$

□

Proof of Lemma 4.2 (Color Threshold Lemma)

Proof. Let m be an arbitrary odd integer less than k. When $\rho > z_m$, the definition of z_m tells us that there are m colors (which are distinct from each other and not equal to $c = \ell(e)$) that want at least one of the nodes in $e = (u, v)$. By the pigeonhole principle, at least $h = \lceil \frac{m}{2} \rceil = \frac{m+1}{2}$ of these colors want the *same* node. Without loss of generality, let u be this node and $\{c_1, c_2, \dots, c_h\}$ be these h colors, arranged so that

$$x_u^{c_1} \leq x_u^{c_2} \leq \cdots \leq x_u^{c_h} \leq z_m.$$

Without loss of generality we can assume these h colors are the colors (not including c) that want node u the most, in the sense that $x_u^j \geq x_u^{c_h}$ for every color $j \notin \{c, c_1, c_2, \dots, c_h\}$. Meanwhile, for every $\rho \leq z_m$, there can be at most $m - h = \frac{m-1}{2}$ *other* colors not in $\{c, c_1, c_2, \dots, c_h\}$ that want node v. We can show then that for $j \in \{1, 2, \dots, h\}$,

$$x_u^{c_j} \leq z_{j+m-h}. \tag{9}$$

If this were not true and we instead assume $x_u^{c_j} > z_{j+m-h}$, this would mean that for any $\rho \in (z_{j+m-h}, x_u^{c_j})$, there are $j + m - h$ distinct colors (not equal to c) that want at least one of the two nodes in $e = (u, v)$. However, since $\rho < x_u^{c_j}$, we know that $\{c_1, c_2, \dots, c_{j-1}\}$ are the only colors (not counting c) that want node u. Furthermore, since $\rho < z_m$, there are at most $m - h$ *distinct* colors (again not counting c) that want node v. Thus, $\rho < x_u^{c_j}$ would imply there are at most $j + m - h - 1$ colors not equal to c that want either u or v. This is a contradiction, so (9) must hold. Combining this inequality with the fact that $\sum_{i=1}^{k} x_u^i = k - 1$, we obtain

$$h \leq x_u^c + \sum_{j=1}^{h} x_u^{c_j} \leq x_e + \sum_{j=1}^{h} z_{j+m-h} = x_e + z_{1+m-h} + z_{2+m-h} + \cdots + z_m.$$

Since we assumed that m is odd, we know $h = 1 + m - h$ and $m = 2h - 1$, so setting $t = h$ yields the inequality in the statement of the lemma. □

$$
\begin{aligned}
\text{maximize} \quad & \frac{q}{q+1}\frac{7}{8}\chi - \sum_{j=1}^{q} \frac{1}{j(j+1)}\omega_j \\
\text{subject to} \quad & \textbf{(A\{i\})}\ \omega_i - \omega_{i+1} \leq 0 \text{ for } i = 1, \dots, 5 \\
& \textbf{(A6)}\ \chi - \omega_1 \leq 1 \\
& \textbf{(A7)}\ 2\chi - \omega_2 - \omega_3 \leq 1 \\
& \textbf{(A8)}\ 3\chi - 3\omega_5 \leq 1 \\
& \textbf{(A9)}\ -\chi \leq -2 \\
& \textbf{(A10)}\ \omega_q - \tfrac{7}{8}\chi \leq 0.
\end{aligned}
\tag{12}
$$

Figure 5. This linear program provides an upper bound for $\mathbb{P}\left[e \in \mathcal{M}_Y\right]/x_e$ when $x_e < \frac{1}{2} \leq z_1 \leq z_q \leq \frac{7}{8}$ for integers $q \leq 6$. We explicitly name each constraint as it will be convenient to reference individual constraints in later parts of the proof.

Proofs for Lemmas on Auxiliary Linear Programs

Lemmas 4.3 and 4.4 are concerned with the LPs in Figures 5 and 6. For notational ease in proving these results, let M be the event $e \in \mathcal{M}_Y$, i.e., the event of making a mistake at edge e when rounding the LP relaxation. These Lemmas allow us to bound $\frac{\mathbb{P}[M]}{x_e}$ by solving a small linear program, first for the case $x_e \in (\frac{1}{8}, \frac{1}{2})$, and then for the case $x_e \in \left[\frac{1}{2}, \frac{3}{4}\right)$. Both results will repeatedly apply the following facts:

$$
\mathbb{P}\left[M \mid \rho > x_e \text{ and } \rho \in (z_i, z_{i+1})\right] = \frac{i}{i+1}, \tag{10}
$$

$$
\mathbb{P}\left[\rho \in (a,b)\right] = \frac{8}{3}(b-a), \ \text{for } (a,b) \subseteq \left(\frac{1}{2}, \frac{7}{8}\right). \tag{11}
$$

Proof of Lemma 4.3

Proof. When $x_e < \frac{1}{2}$, we know that for any $\rho \in (\frac{1}{2}, \frac{7}{8})$, the color $c = \ell(e)$ will want both nodes in $e = (u,v)$. If we use Lemma 4.2 with $t = 4$ and the monotonicity of color thresholds from (4), we can see that

$$
4 \leq x_e + z_4 + z_5 + z_6 + z_7 \leq x_e + 4z_7 \implies 1 - \frac{x_e}{4} \leq z_7 \implies z_7 > \frac{7}{8}.
$$

This implies that for a random $\rho \in (\frac{1}{2}, \frac{7}{8})$, at most 6 colors other than c will want a node from $e = (u,v)$. Thus, if $z_{p-1} \leq \frac{1}{2} \leq z_p \leq z_q \leq \frac{7}{8} \leq z_{q+1}$, then $q \leq 6$. Similarly, we can see that $1 \leq x_e + z_1 \implies z_1 > \frac{1}{2}$, so we must have $p = 1$. Next, we use Eq. (10) and Eq. (11) to provide a convenient expression for $\mathbb{P}\left[M\right]$:

$$
\begin{aligned}
\mathbb{P}\left[M\right] &= \sum_{j=1}^{q-1} \mathbb{P}\left[M \mid \rho \in (z_j, z_{j+1})\right] \mathbb{P}\left[\rho \in (z_j, z_{j+1})\right] + \mathbb{P}\left[M \mid \rho \in (z_q, 7/8)\right] \mathbb{P}\left[\rho \in (z_q, 7/8)\right] \\
&= \frac{8}{3}\left(\frac{q}{q+1}\left(\frac{7}{8} - z_q\right) + \sum_{j=1}^{q-1} \frac{j}{j+1}(z_{j+1} - z_j)\right) = \frac{8}{3}\left(\frac{q}{q+1}\frac{7}{8} - \sum_{j=1}^{q} \frac{1}{j(j+1)} z_j\right).
\end{aligned}
$$

Our goal is to upper bound $\frac{\mathbb{P}[M]}{x_e}$. To do so we have the following inequalities and case-specific assumptions at our disposal: (1) the monotonicity of color thresholds: $z_i \leq z_{i+1}$ for $i \in \{0, 1, \dots, k\}$, (2) the relationship between x_e and color thresholds given in Lemma 4.2, and (3) our assumptions that $x_e < \frac{1}{2}$ and $z_q \leq \frac{7}{8} \leq z_{q+1}$. Since at most 6 colors other than c can want a node in $e = (u,v)$, we can extract a set of 10 convenient inequalities from these three categories that we know

$$
\begin{aligned}
\text{max} \quad & \frac{1}{p} + \left(\frac{q}{q+1}\frac{7}{8} - \frac{1}{2}\right)\chi - \sum_{j=p}^{q} \frac{1}{j(j+1)}\omega_j \\
\text{s.t.} \quad & \textbf{(B\{i\})}\ \omega_i - \omega_{i+1} \leq 0 \text{ for } i = 1, \dots, 9 \\
& \textbf{(B10)}\ \chi - \omega_1 \leq 1 \\
& \textbf{(B11)}\ 2\chi - \omega_2 - \omega_3 \leq 1 \\
& \textbf{(B12)}\ 3\chi - \omega_3 - \omega_4 - \omega_5 \leq 1 \\
& \textbf{(B13)}\ 4\chi - 4\omega_7 \leq 1 \\
& \textbf{(B14)}\ \omega_{p-1} \leq 1 \\
& \textbf{(B15)}\ -\omega_p \leq -1 \\
& \textbf{(B16)}\ \omega_q - \tfrac{7}{8}\chi \leq 0.
\end{aligned}
\tag{13}
$$

Figure 6. This linear program provides an upper bound for $\mathbb{P}\left[e \in \mathcal{M}_Y\right]/x_e$ when $x_e \in [\frac{1}{2}, \frac{3}{4})$ and $z_{p-1} \leq x_e \leq z_p \leq z_q \leq \frac{7}{8} \leq z_{q+1}$ for integers $p \leq 5$ and $q \leq 10$ satisfying $p \leq q$. We explicitly name each constraint as it will be convenient to reference individual constraints in later parts of the proof. We could add additional constraints based on the assumptions in Lemma 4.4 and the relationship between variables given in Lemma 4.2, but it suffices to consider a subset of these constraints to bound $\mathbb{P}\left[e \in \mathcal{M}_Y\right]/x_e$, and this also simplifies the analysis.

are true for x_e and the first 6 color threshold values:

Monotonicity Constraints

$$\textbf{(A1)}\ \frac{z_1}{x_e} \leq \frac{z_2}{x_e},\ \textbf{(A2)}\ \frac{z_2}{x_e} \leq \frac{z_3}{x_e},\ \textbf{(A3)}\ \frac{z_3}{x_e} \leq \frac{z_4}{x_e},\ \textbf{(A4)}\ \frac{z_4}{x_e} \leq \frac{z_5}{x_e},\ \textbf{(A5)}\ \frac{z_5}{x_e} \leq \frac{z_6}{x_e}$$

Edge-Node Relationship Constraints (Lemma 4.2)

$$\textbf{(A6)}\ \frac{1}{x_e} \leq 1 + \frac{z_1}{x_e},\ \textbf{(A7)}\ \frac{2}{x_e} \leq 1 + \frac{z_2}{x_e} + \frac{z_3}{x_e},\ \textbf{(A8)}\ \frac{3}{x_e} \leq 1 + \frac{3z_5}{x_e}$$

Boundary Assumption Constraints

$$\textbf{(A9)}\ 1 \leq \frac{1}{2}\frac{1}{x_e},\ \textbf{(A10)}\ \frac{z_q}{x_e} \leq \frac{7}{8}\frac{1}{x_e}.$$

If the maximum value of $\frac{\mathbb{P}[M]}{x_e}$ over all values of $\{x_e, z_1, z_2, z_3, z_4, z_5, z_6\}$ that respect these inequalities is at most P, then this guarantees $\mathbb{P}\left[M\right] \leq P x_e$. Finally, note that $\frac{\mathbb{P}[M]}{x_e}$ and the inequalities above can be given as linear expressions in terms of $\frac{1}{x_e}$ and $\left\{\frac{z_i}{x_e} : i \in \{1, 2, 3, 4, 5, 6\}\right\}$. We can therefore maximize $\frac{3}{8}\frac{\mathbb{P}[M]}{x_e}$ subject to these inequalities by setting $\chi = \frac{1}{x_e}$ and $\omega_i = \frac{z_i}{x_e}$ and solving the linear program in Figure 5. Note that we choose to maximize $\frac{3}{8}\frac{\mathbb{P}[M]}{x_e}$, rather than simply maximizing $\frac{\mathbb{P}[M]}{x_e}$, only because this will simplify some of our analysis later. □

Proof of Lemma 4.4

Proof. Combining Lemma 4.2 (with $t = 6$), the inequalities in (4), and the bound $x_e < \frac{3}{4}$, we have

$$6 \leq x_e + \sum_{j=6}^{11} z_j \leq x_e + 6z_{11} \implies 1 - \frac{x_e}{6} \leq z_{11} \implies z_{11} > \frac{7}{8}.$$

Thus, for every $\rho \in (\frac{1}{2}, \frac{7}{8})$ we know $q \leq 10$ and there are at most 10 colors other than $c = \ell(e)$ that want a node in e. Using similar steps we can show that $z_5 > \frac{3}{4} > x_e$, so $p \leq 5$. We again use Eq. (10) and Eq. (11) to derive the following useful expression for $\mathbb{P}[M]$:

$$\begin{aligned}
\mathbb{P}[M] &= \mathbb{P}\left[M \mid \rho \in \left(\frac{1}{2}, x_e\right)\right]\mathbb{P}\left[\rho \in \left(\frac{1}{2}, x_e\right)\right] + \mathbb{P}[M \mid \rho \in (x_e, z_p)]\,\mathbb{P}[\rho \in (x_e, z_p)] \\
&\quad + \left(\sum_{j=p}^{q-1} \mathbb{P}[M \mid \rho \in (z_j, z_{j+1})]\,\mathbb{P}[\rho \in (z_j, z_{j+1})]\right) + \mathbb{P}\left[M \mid \rho \in \left(z_q, \frac{7}{8}\right)\right]\mathbb{P}\left[\rho \in \left(z_q, \frac{7}{8}\right)\right] \\
&= \frac{8}{3}\left(\left(x_e - \frac{1}{2}\right) + \frac{p-1}{p}(z_p - x_e) + \left(\sum_{j=p}^{q-1} \frac{j}{j+1}(z_{j+1} - z_j)\right) + \frac{q}{q+1}\left(\frac{7}{8} - z_q\right)\right) \\
\implies \frac{\mathbb{P}[M]}{x_e} &\leq \frac{8}{3}\left(\frac{1}{p} + \left(\frac{q}{q+1}\frac{7}{8} - \frac{1}{2}\right)\frac{1}{x_e} - \sum_{j=p}^{q} \frac{1}{j(j+1)}\frac{z_j}{x_e}\right).
\end{aligned}$$

We apply the monotonicity inequalities (4) for color thresholds, Lemma 4.2, and our assumptions in the statement of the lemma to obtain a set of inequalities that can be used to bound $\frac{\mathbb{P}[M]}{x_e}$.

$$\textbf{(B\{i\})}\ \frac{z_i}{x_e} \leq \frac{z_{i+1}}{x_e} \text{ for } i = 1, 2, \ldots 9,$$

$$\textbf{(B10)}\ \frac{1}{x_e} \leq 1 + \frac{z_1}{x_e},\ \textbf{(B11)}\ \frac{2}{x_e} \leq 1 + \frac{z_2}{x_e} + \frac{z_3}{x_e},\ \textbf{(B12)}\ \frac{3}{x_e} \leq 1 + \sum_{j=3}^{5} \frac{z_j}{x_e},$$

$$\textbf{(B13)}\ \frac{4}{x_e} \leq 1 + 4\frac{z_7}{x_e},\ \textbf{(B14)}\ \frac{z_{p-1}}{x_e} \leq 1,\ \textbf{(B15)}\ 1 \leq \frac{z_p}{x_e},\ \textbf{(B16)}\ \frac{z_q}{x_e} \leq \frac{7}{8}\frac{1}{x_e}.$$

We can maximize $\frac{3}{8}\frac{\mathbb{P}[M]}{x_e}$ subject to these inequalities by solving the linear program in Figure 6. □

Proofs not contained in this section of the appendix The proof of Theorem 2.2 is given in Appendix D, which provides additional details on the relationship between MINECC and related multiway cut objectives. Details for proving Lemma 4.5, and in particular dual LP solutions for 46 auxiliary linear programs needed to prove this result, are contained in Appendix C.

C. Optimal Dual Variables and LP bounds for $r = 2$

In Section 4.4 of the main text we proved that when $q = 1$, the solution to the linear program in Figure 5 is bounded above by $\frac{1}{2}$. We accomplished this by multiplying constraints in this LP by dual variables corresponding to these constraints. In Table 3, we present dual variables for all cases $q \in \{1, 2, 3, 4, 5, 6\}$. For the sake of completeness, here we provide additional details for how to prove an upper bound on the LP solution for all of these values of q, without having to explicitly write down and sum a linear combination of constraints. The upper bound can be shown more succinctly by checking a few matrix-vector products, which is the approach we take here. We also provide dual variables for the LP in Figure 6 for all valid choices of p and q, which can be used to prove the desired upper bound on this LP in the same fashion.

C.1. Overview of LP bounding strategy

The linear programs in Figures 5 and 6 can both be written in the following format:

$$\begin{array}{ll} \max & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} & \mathbf{A}\mathbf{x} \leq \mathbf{b}. \end{array} \tag{14}$$

The dual of this LP is given by

$$\begin{array}{ll} \min & \mathbf{b}^T\mathbf{y} \\ \text{s.t.} & \mathbf{A}^T\mathbf{y} = \mathbf{c} \\ & \mathbf{y} \geq 0. \end{array} \tag{15}$$

We want to prove that $\mathbf{c}^T\mathbf{x}^* \leq \frac{1}{2}$ where $\mathbf{x}^*$ is the optimal solution to the primal LP (14). By weak duality theory, it is sufficient to find a vector $\mathbf{y}$ of dual variables satisfying the constraints in the dual LP (15), such that $\mathbf{b}^T\mathbf{y} \leq \frac{1}{2}$. Note that our

Table 3. Optimal dual variables (one for each constraint in the primal LP) for the linear program in Figure 5, for each valid integer q. Multiplying dual variables by the left and right hand sides of each constraint and adding the left and right hand sides together proves the upper bound (last column) on the optimal solution to the primal LP. The bound is always $\frac{1}{2}$ or smaller.

	$\omega_1 - \omega_2 \leq 0$	$\omega_2 - \omega_3 \leq 0$	$\omega_3 - \omega_4 \leq 0$	$\omega_4 - \omega_5 \leq 0$	$\omega_5 - \omega_6 \leq 0$	$\chi - \omega_1 \leq 1$	$2\chi - \omega_2 - \omega_3 \leq 1$	$3\chi - 3\omega_5 \leq 1$	$-\chi \leq -2$	$\omega_q - \frac{7}{8}\chi \leq 0$	
q	**(A1)**	**(A2)**	**(A3)**	**(A4)**	**(A5)**	**(A6)**	**(A7)**	**(A8)**	**(A9)**	**(A10)**	Bound
1	0	0	0	0	0	$\frac{1}{2}$	0	0	$\frac{1}{16}$	0	$\frac{3}{8}$
2	$\frac{1}{6}$	0	0	0	0	$\frac{2}{3}$	0	0	$\frac{1}{12}$	0	$\frac{1}{2}$
3	0	0	0	0	0	$\frac{1}{2}$	$\frac{1}{6}$	0	$\frac{5}{48}$	$\frac{1}{12}$	$\frac{11}{24}$
4	0	0	$\frac{1}{12}$	0	0	$\frac{1}{2}$	$\frac{1}{6}$	0	$\frac{5}{48}$	$\frac{1}{30}$	$\frac{11}{24}$
5	0	0	$\frac{1}{12}$	$\frac{1}{30}$	0	$\frac{1}{2}$	$\frac{1}{6}$	0	$\frac{5}{48}$	0	$\frac{11}{24}$
6	0	0	$\frac{1}{12}$	$\frac{1}{30}$	$\frac{1}{42}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{126}$	$\frac{3}{28}$	0	$\frac{29}{63}$

goal is actually to prove that this can be done for multiple slight variations of the objective function vector $\mathbf{c}$ and constraint matrix $\mathbf{A}$. For all cases we have the same right hand side vector $\mathbf{b}$. If $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_j\}$ denotes the set of objective function vectors, $\{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_j\}$ is the set of corresponding constraint matrices, and $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_j\}$ is an appropriately chosen set of dual vectors, then for $i = 1, 2, \ldots, j$, we need to perform one matrix-vector product and one vector inner product to confirm that

$$\mathbf{A}_j^T \mathbf{y}_j = \mathbf{c}_j$$
$$\mathbf{b}^T \mathbf{y}_j \leq \frac{1}{2}.$$

C.2. Matrix computations for bounding LP in Figure 5

For the linear program in Figure 5, the variable vector is $\mathbf{x}^T = \begin{bmatrix}\omega_1 & \omega_2 & \omega_3 & \omega_4 & \omega_5 & \omega_6 & \chi\end{bmatrix}$ and the right hand size vector is $\mathbf{b}^T = \begin{bmatrix}0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & -2 & 0\end{bmatrix}$. The dual variables we will use are given in Table 3. For each $q \in \{1, 2, \ldots 6\}$, let $\mathbf{y}_q$ denote the vector of dual variables (which corresponds to the qth row in Table 3), and let $\mathbf{Y} = \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_6\end{bmatrix}$, so

$$\mathbf{Y} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{16} & 0 \\ \frac{1}{6} & 0 & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & \frac{1}{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} & \frac{1}{12} \\ 0 & 0 & \frac{1}{12} & 0 & 0 & \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} & \frac{1}{30} \\ 0 & 0 & \frac{1}{12} & \frac{1}{30} & 0 & \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} & 0 \\ 0 & 0 & \frac{1}{12} & \frac{1}{30} & \frac{1}{42} & \frac{1}{2} & \frac{1}{6} & \frac{1}{126} & \frac{3}{28} & 0 \end{bmatrix}.$$

In order to check that $\mathbf{b}^T\mathbf{y}_q \leq \frac{1}{2}$ for each choice of q we just need to compute $\mathbf{Yb}$, which is made easier by the fact that most entries of $\mathbf{b}$ are zero.

$$\mathbf{Yb} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{16} \\ \frac{2}{3} & 0 & 0 & \frac{1}{12} \\ \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} \\ \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} \\ \frac{1}{2} & \frac{1}{6} & 0 & \frac{5}{48} \\ \frac{1}{2} & \frac{1}{6} & \frac{1}{126} & \frac{3}{28} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -2 \end{bmatrix} = \begin{bmatrix} \frac{3}{8} \\ \frac{1}{2} \\ \frac{11}{24} \\ \frac{11}{24} \\ \frac{11}{24} \\ \frac{29}{63} \end{bmatrix}.$$

This confirms that as long as these dual variables are feasible for the dual linear program, the primal LP is always bounded above by $\frac{1}{2}$. So, we just need to confirm that $\mathbf{A}_q^T\mathbf{y}_q = \mathbf{c}_q$ for each q.

The constraint matrix $\mathbf{A}_q$ is nearly the same for all choices of q. Only the last constraint ($\omega_q - \frac{7}{8}\chi \leq 0$) is dependent on q. The first nine rows of the constraint matrix are always given by

$$\mathbf{A}_q(1\colon 9, :\,) = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & -1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & -3 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}.$$

The last row has two entries: $\mathbf{A}_q(10, 7) = -\frac{7}{8}$ and $\mathbf{A}_q(10, q) = 1$. The objective function is

$$\mathbf{c}_q^T\mathbf{x} = \frac{7q}{8(q+1)}\chi - \sum_{j=1}^{q} \frac{1}{j(j+1)}\omega_j,$$

where $\mathbf{c}_q$ is the objective function vector for $q \in \{1, 2, \dots, 6\}$. The matrix $\mathbf{C} = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_6 \end{bmatrix}$ of objective function vectors is given by:

$$\mathbf{C} = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ 0 & -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} \\ 0 & 0 & -\frac{1}{12} & -\frac{1}{12} & -\frac{1}{12} & -\frac{1}{12} \\ 0 & 0 & 0 & -\frac{1}{20} & -\frac{1}{20} & -\frac{1}{20} \\ 0 & 0 & 0 & 0 & -\frac{1}{30} & -\frac{1}{30} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{42} \\ \frac{7}{16} & \frac{7}{12} & \frac{21}{32} & \frac{7}{10} & \frac{35}{48} & \frac{3}{4} \end{bmatrix}$$

Given the matrices and vectors listed above, a few simple computations confirm that $\mathbf{A}_q^T\mathbf{y}_q = \mathbf{c}_q$ indeed holds for every $q \in \{1, 2, \dots 6\}$.

C.3. Dual variables for LP in Figure 6

Tables 4 and 5 give optimal dual variables for the linear program in Figure 6 for all possible values of p and q. We also list the optimal solution to each linear program for each of the 40 cases, which is always less than or equal to $\frac{1}{2}$. We highlight again that our proof does not rely on proving that this value is optimal—it suffices to use the dual variables to get a linear combination of inequalities such that summing the left hand sides produces the LP objective function, and summing the right hand sides yields the LP upper bound given in the last column of each row. We can use the strategy outlined in the previous subsection to check these dual variables and prove the upper bound by computing a matrix-vector product and a vector inner product for each case. We omit detailed calculations as they are straightforward but would take up a significant amount of space.

C.4. Challenges in Avoiding Case Analysis

It is natural to wonder whether we can rule out or simultaneously handle a large number of cases for Lemmas 4.3 and 4.4, rather than check a different set of dual variables for 46 small linear programs. However, there are several challenges in escaping from this lengthy case analysis. First of all, the 46 linear programs we have considered all have feasible solutions, so we cannot rule any out on the basis that they they correspond to impossible cases. The optimal solution for many cases is exactly $\frac{1}{2}$, so the inequalities we apply to bound $\frac{\mathbb{P}[M]}{x_e}$ for these cases must be tight. For many of the other cases, the optimal solution is very close to $\frac{1}{2}$. We attempted to use looser bounds to prove a $\frac{1}{2}$ bound for multiple cases at once, but were unsuccessful. Each case seems to require a slightly different argument in order to prove the needed bound. Adding more constraints to the linear programs also did not lead to a simpler analysis.

A second natural strategy is to try to first identify and prove which values of x_e, p, and q correspond to the worst cases, and then simply confirm that the probability bound holds for these cases. However, the worst case values for x_e, p, and q are not obvious, and are in fact somewhat counterintuitive. At first glance it may appear that the worse case is when q is as large as possible and p is as small as possible. This maximizes the expected number of colors that want a node in e for a randomly chosen $\rho \in (\frac{1}{2}, \frac{7}{8})$, leading to a higher probability of making a mistake at e. However it is important to recall that the goal is to bound $\frac{\mathbb{P}[M]}{x_e}$, and not just $\mathbb{P}[M]$, so this line of reasoning does not apply. As it turns out, the worst case scenario (i.e., largest optimal LP solution) for $x_e \in [\frac{1}{2}, \frac{3}{4})$ is when $q \in \{2, 4\}$, but we do not have a proof (other than by checking all cases) for why this case leads to the largest value of $\frac{\mathbb{P}[M]}{x_e}$.

Finally, another natural question is whether we can avoid a lengthy case analysis by choosing an interval other than $I = (\frac{1}{2}, \frac{7}{8})$ when applying Algorithm 1. It is especially tempting to use an interval (a, b) where $b < \frac{7}{8}$, since this would potentially decrease the maximum number of colors that want a node in $e = (u, v)$. This is desirable because if fewer colors want a node in e, then we have fewer color threshold variables to worry about and fewer cases to consider. The following result (Lemma 4.7 in the main text) shows why this approach will fail.

Lemma C.1. *Let Y denote the node color map returned by running Algorithm 1 with $I = (a, b) \subseteq (0, 1)$ on an edge-colored graph.*

- *If $a > \frac{1}{2}$, there exists a feasible LP solution such that $\mathbb{P}[e \in \mathcal{M}_Y] > \frac{4}{3}x_e$.*
- *If $a \leq \frac{1}{2}$ and $b < \frac{7}{8}$, there exists a feasible LP solution such that $\mathbb{P}[e \in \mathcal{M}_Y] > \frac{4}{3}x_e$.*

Proof. If $a > \frac{1}{2}$, then there exists some $\varepsilon > 0$ such that $a = \frac{1+\varepsilon}{2}$. Consider an edge $e = (u, v)$ with color $c = \ell(e)$ whose LP variables satisfy $x_e = x_u^c = x_v^c = \frac{1-\varepsilon}{2}$ and where $x_u^i = x_v^j = \frac{1+\varepsilon}{2} = a$ for two colors $i \neq j$ that are both distinct from c. This is feasible as long all as $x_u^t = 1$ for $t \notin \{c, i\}$ and $x_v^t = 1$ for $t \notin \{c, j\}$. For every $\rho \in (a, b)$, color i will want node u, color j will want node v, and color c will want both of them. Therefore, the probability of making a mistake at e is exactly

$$\mathbb{P}[e \in \mathcal{M}_Y] = \frac{2}{3} = \frac{2}{3} \cdot \frac{1}{x_e} x_e = \frac{2}{3} \cdot \frac{2}{1-\varepsilon} x_e > \frac{4}{3} x_e.$$

If instead $a \leq \frac{1}{2}$, consider the feasible solution where $x_e = x_u^c = x_v^c = x_u^i = x_u^g = x_v^j = x_v^h = \frac{2}{3}$ where $\{c, g, h, i, j\}$ are all distinct colors. If $b < \frac{2}{3}$ we are guaranteed to make a mistake at x_e, so assume that $\frac{2}{3} \leq b < \frac{7}{8}$. Then we can calculate that

$$\frac{\mathbb{P}[e \in \mathcal{M}_Y]}{x_e} = \frac{1}{b-a}\left(\frac{2}{3} - a + \frac{4}{5}\left(b - \frac{2}{3}\right)\right) \cdot \frac{3}{2} = \frac{2-3a}{10(b-a)} + \frac{6}{5} > \frac{2 - 3 \cdot \frac{1}{2}}{10(\frac{7}{8} - \frac{1}{2})} + \frac{6}{5} = \frac{4}{3}.$$

□

The first case in this lemma indicates that we should not use an interval (a, b) where $a > \frac{1}{2}$. The second case rules out the hope of choosing some $b < \frac{7}{8}$ in order to make the analysis easier. The interval $I = (\frac{1}{2}, \frac{7}{8})$ is chosen to avoid these two issues and be as simple to analyze as possible.

Table 4. Optimal dual variables, one for each of the 16 constraints, for the linear program in Figure 6, for each valid pair (p, q) where $p \in \{1, 2\}$. We omit the column for constraint number 6, $\omega_6 - \omega_7 \leq 0$, since the dual variable for this constraint is always zero for these cases. The last column is the bound on the LP objective (it is in fact the optimal LP solution value) that we can prove for each case using the dual variables. It is always $\frac{1}{2}$ or less.

Constraints	$\omega_1 - \omega_2 \leq 0$	$\omega_2 - \omega_3 \leq 0$	$\omega_3 - \omega_4 \leq 0$	$\omega_4 - \omega_5 \leq 0$	$\omega_5 - \omega_6 \leq 0$	$\omega_7 - \omega_8 \leq 0$	$\omega_8 - \omega_9 \leq 0$	$\omega_9 - \omega_{10} \leq 0$	$\chi - \omega_1 \leq 1$	$2\chi - \omega_2 - \omega_3 \leq 1$	$3\chi - \omega_3 - \omega_4 - \omega_5 \leq 1$	$4\chi - 4\omega_7 \leq 1$	$\omega_{p-1} \leq 1$	$-\omega_p \leq -1$	$\omega_q - \frac{7}{8}\chi \leq 0$	LP bound
p,q	**1**	**2**	**3**	**4**	**5**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	
1, 1	0	0	0	0	0	0	0	0	0	0	0	0	0	$\frac{4}{7}$	$\frac{1}{14}$	$\frac{3}{7}$
2	$\frac{1}{6}$	0	0	0	0	0	0	0	$\frac{1}{12}$	0	0	0	0	$\frac{7}{12}$	0	$\frac{1}{2}$
3	$\frac{3}{32}$	$\frac{1}{192}$	0	0	0	0	0	0	0	$\frac{5}{64}$	0	0	0	$\frac{19}{32}$	0	$\frac{31}{64}$
4	$\frac{1}{10}$	$\frac{1}{30}$	$\frac{1}{20}$	0	0	0	0	0	0	$\frac{1}{10}$	0	0	0	$\frac{3}{5}$	0	$\frac{1}{2}$
5	$\frac{47}{450}$	0	$\frac{13}{900}$	0	0	0	0	0	0	$\frac{14}{225}$	$\frac{8}{225}$	0	0	$\frac{136}{225}$	$\frac{1}{450}$	$\frac{37}{75}$
6	$\frac{3}{28}$	0	$\frac{5}{252}$	$\frac{17}{1260}$	$\frac{1}{42}$	0	0	0	0	$\frac{5}{84}$	$\frac{11}{252}$	0	0	$\frac{17}{28}$	0	$\frac{125}{252}$
7	$\frac{7}{64}$	0	$\frac{37}{2016}$	$\frac{257}{20160}$	$\frac{1}{42}$	0	0	0	0	$\frac{11}{192}$	$\frac{179}{4032}$	$\frac{1}{224}$	0	$\frac{39}{64}$	0	$\frac{2003}{4032}$
8	$\frac{1}{9}$	0	$\frac{13}{756}$	$\frac{23}{1890}$	$\frac{1}{42}$	$\frac{1}{72}$	0	0	0	$\frac{1}{18}$	$\frac{17}{378}$	$\frac{1}{126}$	0	$\frac{11}{18}$	0	$\frac{94}{189}$
9	$\frac{9}{80}$	0	$\frac{41}{2520}$	$\frac{59}{5040}$	$\frac{1}{42}$	$\frac{1}{40}$	$\frac{1}{90}$	0	0	$\frac{13}{240}$	$\frac{229}{5040}$	$\frac{3}{280}$	0	$\frac{49}{80}$	0	$\frac{2509}{5040}$
10	$\frac{5}{44}$	0	$\frac{43}{2772}$	$\frac{157}{13860}$	$\frac{1}{42}$	$\frac{3}{88}$	$\frac{2}{99}$	$\frac{1}{110}$	0	$\frac{7}{132}$	$\frac{127}{2772}$	$\frac{1}{77}$	0	$\frac{27}{44}$	0	$\frac{1381}{2772}$
2, 2	0	0	0	0	0	0	0	0	$\frac{1}{12}$	0	0	0	$\frac{1}{12}$	$\frac{1}{6}$	0	$\frac{1}{2}$
3	0	$\frac{1}{192}$	0	0	0	0	0	0	0	$\frac{5}{64}$	0	0	0	$\frac{3}{32}$	0	$\frac{31}{64}$
4	0	$\frac{1}{30}$	$\frac{1}{20}$	0	0	0	0	0	0	$\frac{1}{10}$	0	0	0	$\frac{1}{10}$	0	$\frac{1}{2}$
5	0	0	$\frac{13}{900}$	0	0	0	0	0	0	$\frac{14}{225}$	$\frac{8}{225}$	0	0	$\frac{47}{450}$	$\frac{1}{450}$	$\frac{37}{75}$
6	0	0	$\frac{5}{252}$	$\frac{17}{1260}$	$\frac{1}{42}$	0	0	0	0	$\frac{5}{84}$	$\frac{11}{252}$	0	0	$\frac{3}{28}$	0	$\frac{125}{252}$
7	0	0	$\frac{37}{2016}$	$\frac{257}{20160}$	$\frac{1}{42}$	0	0	0	0	$\frac{11}{192}$	$\frac{179}{4032}$	$\frac{1}{224}$	0	$\frac{7}{64}$	0	$\frac{2003}{4032}$
8	0	0	$\frac{13}{756}$	$\frac{23}{1890}$	$\frac{1}{42}$	$\frac{1}{72}$	0	0	0	$\frac{1}{18}$	$\frac{17}{378}$	$\frac{1}{126}$	0	$\frac{1}{9}$	0	$\frac{94}{189}$
9	0	0	$\frac{41}{2520}$	$\frac{59}{5040}$	$\frac{1}{42}$	$\frac{1}{40}$	$\frac{1}{90}$	0	0	$\frac{13}{240}$	$\frac{229}{5040}$	$\frac{3}{280}$	0	$\frac{9}{80}$	0	$\frac{2509}{5040}$
10	0	0	$\frac{43}{2772}$	$\frac{157}{13860}$	$\frac{1}{42}$	$\frac{3}{88}$	$\frac{2}{99}$	$\frac{1}{110}$	0	$\frac{7}{132}$	$\frac{127}{2772}$	$\frac{1}{77}$	0	$\frac{5}{44}$	0	$\frac{1381}{2772}$

Table 5. Optimal dual variables, one for each of the 16 constraints, for the linear program in Figure 6, for each pair (p, q) when $p \in \{3, 4, 5\}$. We omit columns for constraint 1 ($\omega_1 - \omega_2 \leq 0$) and constraint 10 ($\chi - \omega_1 \leq 1$), as dual variables for these are always zero for these cases.

Constraints	$\omega_2 - \omega_3 \leq 0$	$\omega_3 - \omega_4 \leq 0$	$\omega_4 - \omega_5 \leq 0$	$\omega_5 - \omega_6 \leq 0$	$\omega_6 - \omega_7 \leq 0$	$\omega_7 - \omega_8 \leq 0$	$\omega_8 - \omega_9 \leq 0$	$\omega_9 - \omega_{10} \leq 0$	$2\chi - \omega_2 - \omega_3 \leq 1$	$3\chi - \omega_3 - \omega_4 - \omega_5 \leq 1$	$4\chi - 4\omega_7 \leq 1$	$\omega_{p-1} \leq 1$	$-\omega_p \leq -1$	$\omega_q - \frac{7}{8}\chi \leq 0$	LP bound
p, q	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**11**	**12**	**13**	**14**	**15**	**16**	
3, 3	0	0	0	0	0	0	0	0	$\frac{5}{64}$	0	0	$\frac{5}{64}$	$\frac{1}{192}$	0	$\frac{31}{64}$
4	0	$\frac{1}{20}$	0	0	0	0	0	0	$\frac{1}{10}$	0	0	$\frac{1}{10}$	$\frac{1}{30}$	0	$\frac{1}{2}$
5	0	$\frac{13}{900}$	0	0	0	0	0	0	$\frac{14}{225}$	$\frac{8}{225}$	0	$\frac{14}{225}$	0	$\frac{1}{450}$	$\frac{37}{75}$
6	0	$\frac{5}{252}$	$\frac{17}{1260}$	$\frac{1}{42}$	0	0	0	0	$\frac{5}{84}$	$\frac{11}{252}$	0	$\frac{5}{84}$	0	0	$\frac{125}{252}$
7	0	$\frac{37}{2016}$	$\frac{257}{20160}$	$\frac{1}{42}$	0	0	0	0	$\frac{11}{192}$	$\frac{179}{4032}$	$\frac{1}{224}$	$\frac{11}{192}$	0	0	$\frac{2003}{4032}$
8	0	$\frac{13}{756}$	$\frac{23}{1890}$	$\frac{1}{42}$	0	$\frac{1}{72}$	0	0	$\frac{1}{18}$	$\frac{17}{378}$	$\frac{1}{126}$	$\frac{1}{18}$	0	0	$\frac{94}{189}$
9	0	$\frac{41}{2520}$	$\frac{59}{5040}$	$\frac{1}{42}$	0	$\frac{1}{40}$	$\frac{1}{90}$	0	$\frac{13}{240}$	$\frac{229}{5040}$	$\frac{3}{280}$	$\frac{13}{240}$	0	0	$\frac{2509}{5040}$
10	0	$\frac{43}{2772}$	$\frac{157}{13860}$	$\frac{1}{42}$	0	$\frac{3}{88}$	$\frac{2}{99}$	$\frac{1}{110}$	$\frac{7}{132}$	$\frac{127}{2772}$	$\frac{1}{77}$	$\frac{7}{132}$	0	0	$\frac{1381}{2772}$
4, 4	$\frac{1}{10}$	0	0	0	0	0	0	0	$\frac{1}{10}$	0	0	$\frac{1}{5}$	$\frac{1}{20}$	0	$\frac{1}{2}$
5	$\frac{3}{64}$	0	0	0	0	0	0	0	$\frac{3}{64}$	$\frac{1}{20}$	0	$\frac{23}{160}$	0	$\frac{1}{60}$	$\frac{157}{320}$
6	$\frac{5}{112}$	0	$\frac{1}{280}$	$\frac{1}{42}$	0	0	0	0	$\frac{5}{112}$	$\frac{3}{56}$	0	$\frac{1}{7}$	0	0	$\frac{55}{112}$
7	$\frac{39}{896}$	0	$\frac{1}{280}$	$\frac{1}{42}$	0	0	0	0	$\frac{39}{896}$	$\frac{3}{56}$	$\frac{1}{224}$	$\frac{9}{64}$	0	0	$\frac{63}{128}$
8	$\frac{43}{1008}$	0	$\frac{1}{280}$	$\frac{1}{42}$	0	$\frac{1}{72}$	0	0	$\frac{43}{1008}$	$\frac{3}{56}$	$\frac{1}{126}$	$\frac{5}{36}$	0	0	$\frac{71}{144}$
9	$\frac{47}{1120}$	0	$\frac{1}{280}$	$\frac{1}{42}$	0	$\frac{1}{40}$	$\frac{1}{90}$	0	$\frac{47}{1120}$	$\frac{3}{56}$	$\frac{3}{280}$	$\frac{11}{80}$	0	0	$\frac{79}{160}$
10	$\frac{51}{1232}$	0	$\frac{1}{280}$	$\frac{1}{42}$	0	$\frac{3}{88}$	$\frac{2}{99}$	$\frac{1}{110}$	$\frac{51}{1232}$	$\frac{3}{56}$	$\frac{1}{77}$	$\frac{3}{22}$	0	0	$\frac{87}{176}$
5, 5	0	$\frac{8}{85}$	0	0	0	0	0	0	0	$\frac{8}{85}$	0	$\frac{16}{85}$	0	$\frac{31}{510}$	$\frac{41}{85}$
6	0	$\frac{8}{85}$	0	$\frac{31}{510}$	0	0	0	0	0	$\frac{8}{85}$	0	$\frac{16}{85}$	0	$\frac{22}{595}$	$\frac{41}{85}$
7	0	$\frac{8}{85}$	0	$\frac{31}{510}$	$\frac{22}{595}$	0	0	0	0	$\frac{8}{85}$	0	$\frac{16}{85}$	0	$\frac{13}{680}$	$\frac{41}{85}$
8	0	$\frac{8}{85}$	0	$\frac{31}{510}$	$\frac{22}{595}$	$\frac{13}{680}$	0	0	0	$\frac{8}{85}$	0	$\frac{16}{85}$	0	$\frac{4}{765}$	$\frac{41}{85}$
9	0	$\frac{3}{32}$	0	$\frac{29}{480}$	$\frac{41}{1120}$	$\frac{1}{40}$	$\frac{1}{90}$	0	0	$\frac{3}{32}$	$\frac{1}{640}$	$\frac{3}{16}$	0	0	$\frac{309}{640}$
10	0	$\frac{41}{440}$	0	$\frac{79}{1320}$	$\frac{111}{3080}$	$\frac{3}{88}$	$\frac{2}{99}$	$\frac{1}{110}$	0	$\frac{41}{440}$	$\frac{7}{1760}$	$\frac{41}{220}$	0	0	$\frac{851}{1760}$

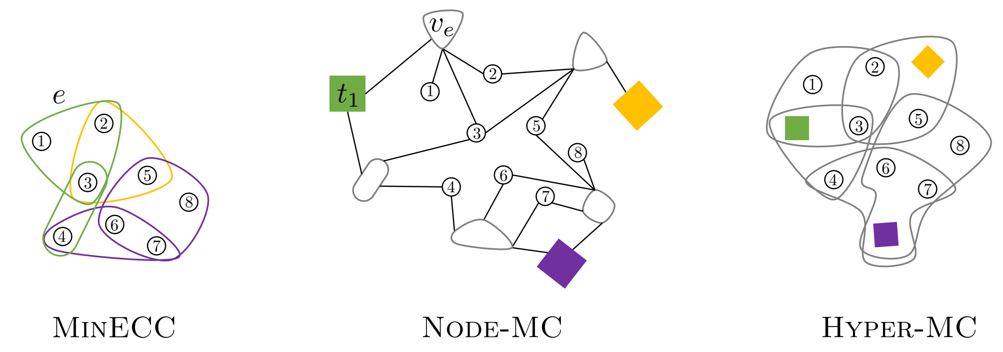

Figure 7. MinECC can be reduced to Node-MC and Hyper-MC in an approximation preserving way. Squares represent terminal nodes, one for each color. In the instance of Node-MC, original nodes (numbered circles) are given weight ∞, so we can only delete hyperedge nodes when separating terminal nodes. In the illustration, green is color 1, and we label one green edge $e = \{1, 2, 3\}$, which is converted to a new node v_e that is attached to terminal t_1 in the Node-MC instance.

Lemma 4.7 and the other challenges highlighted above do not imply that it is impossible to design a $\frac{4}{3}$-approximation algorithm with a simpler approximation guarantee proof. Indeed, a simpler proof of a tight approximation is a compelling direction for future research. Nevertheless, this discussion highlights why it is challenging to escape from using a length case analysis when proving that $\mathbb{P}\left[e \in \mathcal{M}_Y\right] \leq \frac{4}{3}x_e$ for Algorithm 1.

D. Relation to Multiway Cut Objectives

In this section we prove that the canonical MinECC LP relaxation is always at least as tight as the Node-MC LP relaxation, and that it can be strictly tighter in some cases. Therefore, although these approaches lead to a matching worst-case guarantee when r is arbitrarily large, applying the canonical relaxation leads to better results in many cases. We will also highlight why our approximation guarantees for small values of r are much better than any guarantees we obtain by reducing to Hyper-MC.

D.1. Reductions from MinECC to Node-MC and Hyper-MC

As shown previously (Amburg et al., 2020), an instance $H = (V, E, C, \ell)$ of MinECC can be reduced to an instance of Node-MC on a graph G via the following steps

- For every node $v \in V$, add a corresponding node v to G with weight ∞.
- For each color $i \in \{1, 2, \dots, k\}$, place a terminal node t_i in G.
- For each hyperedge $e \in E$ with weight w_e, add a node v_e to G with weight w_e and place an edge (v, v_e) for each $v \in e$.

The MinECC objective on H is then approximation equivalent to Node-MC on G. Node-MC is in turn approximation equivalent to Hyper-MC (Chekuri & Ene, 2011b). Although not previously shown explicitly, there is a particularly simple reduction from MinECC to Hyper-MC: introduce a terminal node t_i for each color i, and then for a hyperedge $e \in E$ of color i, add terminal node t_i to that hyperedge and then ignore the hyperedge color. Figure 7 illustrates the procedure of reducing an instance of MinECC to Node-MC and Hyper-MC. We note the following simple equivalence result.

Observation 3. MinECC with k colors is approximation-equivalent to a special type of Hyper-MC problem on k terminal nodes where every hyperedge contains a terminal node.

Approximation guarantees for MinECC via reductions There are a few known approximation guarantees for Hyper-MC in terms of the maximum hyperedge size r, but these do not imply any useful results for MinECC. When $r = 2$, Hyper-MC is the well-studied graph multiway cut objective (Dahlhaus et al., 1994; Călinescu et al., 2000), but this has no direct bearing on MinECC since an instance of MinECC with maximum hyperedge size $r \geq 2$ corresponds to an instance of Hyper-MC with maximum hyperedge size $r + 1 \geq 3$. Chekuri & Ene (2011b) gave an H_r approximation for

Node-MC LP: path constraints version

$$\boxed{\begin{array}{lll} \min & \sum_{v \in V-T} w_v d_v & \\ \text{s.t.} & \sum_{v\in p} d_v \geq 1 & \forall p \in \mathcal{P} \\ & d_v = 0 & \forall v \in T \\ & d_v \geq 0 & \forall v \in V-T \end{array}} \tag{16}$$

Node-MC LP: polynomial constraints version

$$\boxed{\begin{array}{lll} \min & \sum_{v \in V-T} w_v d_v & \\ \text{s.t.} & y_v^i \leq y_u^i + d_v & \forall (u,v) \in E, \forall i \in [k] \\ & y_{t_i}^i = 0 & \forall i \in [k] \\ & y_{t_i}^j \geq 1 & \forall i,j \in [k], j \neq i \\ & d_v \geq 0 & \forall v \in V-T \end{array}} \tag{17}$$

Figure 8. Two equivalent ways to write the distance-based LP relaxation for node-weighted multiway cut (Garg et al., 2004). $\mathcal{P}$ is the set of all paths between terminal nodes.

HYPER-MC where H_i is the ith harmonic number. However, this only implies a 1.833-approximation for MINECC when $r = 3$, and is worse than a 2-approximation when $r \geq 4$. The approximation guarantees we obtain for small values of r by rounding the MINECC LP relaxation are therefore significantly better and than results obtained by reducing to generalized multiway cut objectives. In contrast, when r is arbitrarily large, our approximation guarantee of $2(1 - \frac{1}{k})$ for rounding the MINECC LP relaxation matches the guarantee obtained by first reducing to NODE-MC and rounding the LP relaxation for this objective (Garg et al., 2004). In this appendix we explore the relationship between these linear programs in more depth.

D.2. The MINECC LP is Tighter than the NODE-MC LP

Let $G = (V, E)$ be a graph with terminal nodes $T = \{t_1, t_2, \dots, t_k\}$ and a weight $w_v \geq 0$ for each node $v \in V$. The distance-based LP relaxation for the NODE-MC objective on G is shown in Figure 8, where $\mathcal{P}$ represents the set of all paths between pairs of terminal nodes. The formulation in (16) uses a single variable d_v for each node v, with the constraint that the distance between every pair of terminal nodes is at least 1. This requires an exponential number of path constraints. The bottom of Figure 8 shows an alternative way to write the LP using more variables but only polynomially many constraints and variables. The two LPs are known to be equivalent (Garg et al., 2004). The variable d_u provides an indication for how strongly we wish to delete node u, and the variable y_u^i is interpreted as the distance between node u and cluster i.

Consider now an instance of MINECC encoded by an edge-colored hypergraph $H = (V, E, C, \ell)$ that we reduce to a node-weighted graph $G = (\hat{V}, \hat{E})$ using the strategy in Section D.1. The node set $\hat{V} = T \cup V \cup V_E$ is made up of terminal nodes $T = \{t_i : i = 1, 2, \dots, k\}$, the original node set V from hypergraph H, and the node set $V_E = \{v_e : e \in E\}$. In Figure 7, these are represented by square nodes, numbered circular nodes, and irregular shaped hyperedge-nodes, respectively. The edge set $\hat{E} = E_H \cup E_T$ has two parts, defined by

$$E_H = \{(v, v_e) \colon v \in e \text{ in } H\} \tag{18}$$
$$E_T = \{(t_i, v_e) \colon \ell(e) = i \text{ in } H\}. \tag{19}$$

The node weight for each $u \in V \cup T$ is given by $w_u = \infty$, and the node weight for $v_e \in V_E$ is w_e. Focusing specifically on the formulation shown in (17), we see that the NODE-MC LP shares some similarities with the canonical MINECC relaxation. For example, both linear programs involve one variable for each node-color pair $(u, i) \in V \times C$. However, in terms of the hypergraph $H = (V, E, C, \ell)$, the NODE-MC relaxation involves $O(k|V| + k|E|)$ variables and $O(k \sum_{e\in E} |e|)$ constraints overall, while the MINECC relaxation has $O(k|V| + |E|)$ variables and $O(k|V| + \sum_{e\in E} |e|)$ constraints. Although the two linear programs both have an integrality gap of $2(1 - \frac{1}{k})$, the following theorem proves that the MINECC relaxation will always be at least as tight as the lower bound obtained via the NODE-MC relaxation.

Theorem D.1. *(Theorem 2.2 in the main text) The value of the* NODE-MC *relaxation on $G = (\hat{V}, \hat{E})$ is at most the value of the* MINECC *relaxation on $H = (V, E, C, \ell)$.*

Proof. Let $\mathbf{X} = \{x_v^i, x_e \colon v \in V, e \in E, i \in [k]\}$ denote an arbitrary set of variables for the MINECC LP relaxation for H. Our goal is to use these to construct a set of feasible variables for the NODE-MC LP relaxation on graph $G = (\hat{V}, \hat{E})$ with the same objective score. This means we have to define y_v^i and d_v for each $v \in \hat{V}$ and $i \in [k]$. Since there are multiple different types of nodes in $\hat{V}$, in order to simplify notation we will let $y_e^i = y_{v_e}^i$ and $d_e = d_{v_e}$ when we are considering a node $v_e \in V_E$ that is associated with a hyperedge $e \in E$. Define the following set of variables for the NODE-MC LP:

- Let $d_v = 0$ for $v \in V \cup T$.
- For $v_e \in V_E$, define $y_e^j = \begin{cases} x_e & \text{if } \ell(e) = j \text{ in } H \\ 1 & \text{otherwise.} \end{cases}$
- For $v_e \in V_E$, define $d_e = y_e^c$ where $c = \ell(e)$ in H.
- For $v \in V$, define $y_v^i = x_v^i$ for every $i \in [k]$.
- For $t_i \in T$ where $i \in [k]$, define $y_{t_i}^i = 0$, and $y_{t_i}^j = 1$ whenever $i \neq j$.

The objective for the NODE-MC LP relaxation is then $\sum_{v_e \in V_e} w_e d_e = \sum_{e \in E} w_e x_e$, which matches the MINECC relaxation value. It only remains to check that the following distance constraints hold for all different types of edges $(u, v) \in \hat{E}$ and every color $j \in [k]$:

$$y_v^j \leq y_u^j + d_v \tag{20}$$
$$y_u^j \leq y_v^j + d_u. \tag{21}$$

For $(v, v_e) \in E_H$, if $\ell(e) = c$ then $y_v^c = x_v^c \leq x_e = y_e^c = y_e^c + d_v$, since $d_v = 0$ for $v \in V$. Therefore, constraint (20) holds in this case. Also, when $\ell(e) = c$, constraint (21) holds because $y_e^c = x_e = d_e \leq y_v^c + d_e$. If $(v, v_e) \in E_H$ but we consider a color $i \neq c = \ell(e)$, then constraint (20) is satisfied because $y_v^i = x_v^i \leq 1 = y_e^i = y_e^i + d_v$. Constraint (21) is satisfied as well since $y_e^i = 1 \leq x_v^i + x_v^c \leq x_v^i + x_e = y_v^i + d_e$. Here we have used the fact that $\sum_{i=1}^k x_v^i = 1$ implies $1 \leq x_v^i + x_v^c$. Finally, for an edge $(v_e, t_c) \in E_T$ where $c = \ell(e)$, constraints (20) and (21) become $y_e^j \leq y_{t_c}^j + d_e$ and $y_{t_c}^j \leq y_e^j$. If $j \neq c$, then both of these hold because $y_{t_c}^j = y_e^j = 1$. Meanwhile, if $j = c$, then they follow from $y_e^j = d_e$ and $y_{t_i}^j = 0$. □

Strictly tighter instance. Theorem D.1 shows that the MINECC relaxation is at least as tight of a lower bound as using the NODE-MC relaxation. Although they have the same integrality gap, we can additionally show that the MINECC relaxation can be strictly tighter. Consider color set $C = \{1, 2, 3\}$ and let $H = (V, E)$ be a star graph with center node v_0 and three leaf nodes $\{v_1, v_2, v_3\}$, where edge $e_i = (v_0, v_i)$ has color i for $i = 1, 2, 3$. The optimal MINECC solution will satisfy one of the edges and make mistakes at the two other edges. The MINECC LP relaxation will also have an optimal value of 2 by setting $x_{v_i}^i = 0$ for $i = 1, 2, 3$, and choosing either $x_{v_0}^i = \frac{2}{3}$ for every $i = 1, 2, 3$, or $x_{v_0}^i = 1$ for exactly one $i \in \{1, 2, 3\}$. Meanwhile, if we reduce to an instance of NODE-MC, each terminal node participates in one edge (v_{e_i}, t_i) for $i \in \{1, 2, 3\}$. If we specifically consider the path constraints formulation of the NODE-MC LP in (16), we can see that it is feasible to set $d_{e_i} = \frac{1}{2}$, leading to an optimal LP solution of $\frac{3}{2}$. This matches the worst case integrality gap.

E. Additional Details for Vertex Cover Results

E.1. Reduction proofs

Proof of Theorem 5.1

Proof. See Figure 3. To construct the edge-colored hypergraph H, introduce a node v_{ij} for each $(i, j) \in E$, and for each $u \in V$ define a hyperedge $e_u = \{v_{uj} \colon j \text{ where } (u, j) \in E)\}$. Let each hyperedge be associated with its own unique color. Each node in G corresponds to a hyperedge in H and deleting a hyperedge in H is equivalent to covering a node in G.

Algorithm 2 PittVertexCover(G)

```
Input: $G = (V_G, E_G)$ with node weight $w_v \geq 0$ for each $v \in V_G$
Output: Vertex cover $\mathcal{C} \subseteq V$.
$\mathcal{C} \leftarrow \emptyset$ // Initialize empty vertex cover
for $(u, v) \in E_G$ do
  if $u \notin \mathcal{C}$ and $v \notin \mathcal{C}$ then
    Generate uniform random $\rho \in (0, 1)$
    if $\rho < \frac{w_v}{w_u + w_v}$ then
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{u\}$
    else
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{v\}$
    end if
  end if
end for
Return $\mathcal{C}$
```

Algorithm 3 PittColoring(H)

```
Input: Edge-colored hypergraph $H = (V, E, C, \ell)$ with weight $w(e) \geq 0$ for each $e \in E$
Output: Set $\mathcal{D} \subseteq E$ so that $H = (V, E - \mathcal{D}, C, \ell)$ has no bad edge pairs
$\mathcal{D} \leftarrow \emptyset$
for $v \in V$ do
  $L_E(v) = [v(1) \;\; v(2) \;\; \cdots \;\; v(d_v)]$ // Indices for $v$'s edges, ordered by color
  $f = 1, b = d_v$                // Pointers to front/back of index array
  // Move past all covered bad edge pairs containing $v$
  while $e_{v(b)} \in \mathcal{D}$ and $b > f$ do
    $b \leftarrow b - 1$
  end while
  while $e_{v(f)} \in \mathcal{D}$ and $b > f$ do
    $f \leftarrow f + 1$
  end while
  // Handle uncovered bad edge pairs containing $v$
  while $\ell(v(f)) \neq \ell(v(b))$ do
    Generate $\rho \in (0, 1)$
    if $\rho < \frac{w(v(f))}{w(v(f)) + w(v(b))}$ then
      $\mathcal{D} \leftarrow \mathcal{D} \cup \{e_{v(b)}\}$
      while $e_{v(b)} \in \mathcal{D}$ and $b > f$ do
        // Ignore any deleted edges
        $b \leftarrow b - 1$
      end while
    else
      $\mathcal{D} \leftarrow \mathcal{D} \cup \{e_{v(f)}\}$
      while $e_{v(f)} \in \mathcal{D}$ and $b > f$ do
        // Ignore any deleted edges
        $f \leftarrow f + 1$
      end while
    end if
  end while
end for
Return $\mathcal{D}$
```

Because all hyperedges have different colors, for every pair of overlapping hyperedges (e_u, e_v) we know that either e_u or e_v must be deleted. This is equivalent to covering node u or node v when $(u, v) \in E$. By construction, the degree distribution in G is exactly the hyperedge size distribution in H. $\square$

Proof of Theorem 5.2

Proof. See Figure 3. To construct the graph G, first introduce a node v_e for every hyperedge $e \in E$. If two hyperedges $e \in E$ and $f \in E$ define a bad hyperedge pair ($|e \cap f| > 0$ and $\ell(e) \neq \ell(f)$), then we add an edge (v_e, v_f) in the graph G. If $e \in E$ has weight w_e, we give node v_e weight w_e in G. Deleting a minimum weight set of hyperedges in H to ensure remaining hyperedges of different colors do not overlap is equivalent to deleting a minimum weight set of nodes in G so that the remaining nodes form an independent set (i.e., the VERTEX COVER problem). $\square$

E.2. Details and pseudocode for combinatorial algorithms

Algorithm 2 is pseudocode for Pitt's algorithm (Pitt, 1985) for weighted VERTEX COVER. Algorithm 3 is pseudocode for PittColoring, which is our 2-approximation for MINECC that implicitly applies Pitt's algorithm to the VERTEX COVER instance G that is approximation-equivalent to solving the MINECC objective on an edge colored hypergraph $H = (V, E, C, \ell)$. Importantly, this algorithm never explicitly forms G, but is still able to identify a set of edges in H to delete which correspond to a vertex cover in G. Full details for how to accomplish this is included in the main text, along with justification for the $O(\sum_{e \in E} |e|)$ runtime. The fact that this is a 2-approximation for MINECC (even in the weighted case) follows directly from the fact that this is implicitly providing a valid 2-approximate vertex cover in the (unformed) reduced graph G via Pitt's algorithm.

MatchColoring. Our algorithm MatchColoring is a 2-approximation for the *unweighted* version of MINECC. This algorithm implicitly applies the common strategy of approximating an unweighted VERTEX COVER problem by finding a maximal matching and then adding both endpoints of each edge in the maximal matching to form a cover. For our problem, this corresponds to finding a maximal edge-disjoint set of bad edge pairs, and deleting all edges in those bad edge pairs. This can also be implemented in $O(\sum_e |e|)$ time by carefully iterating through bad edge pairs in the same way that PittColoring does. This approach has the additional advantage of providing a deterministic 2-approximate solution for unweighted MINECC (PittColoring has an *expected* 2-approximation). Additionally, the maximal matching (i.e., maximal edge-disjoint set of bad edge pairs) that it computes provides a lower bound on the MINECC objective, which can be used in practice to check a posteriori approximation guarantees for other methods that do not come with approximation guarantees of their own.

Improved Hybrid algorithm. Our Hybrid algorithm combines the strengths of MatchColoring and MajorityVote. MatchColoring finds a maximal edge-disjoint set of bad hyperedge pairs, which provides a useful lower bound on the optimal MINECC objective. It deletes all edges in this maximal set, which is exactly within a factor 2 of this lower bound. In practice, this often deletes more edges than is strictly necessary, and leads to a large set of nodes that are contained in no remaining edge. From a theoretical perspective, these can be assigned any color and the algorithm will still be a 2-approximation. Hybrid works simply by assigning these nodes to have the color determined by the MajorityVote method (i.e, the edge color that the node participates in the most). This often ends up satisfying many edges that were deleted by MatchColoring even though they did not actually need to be deleted in order to yield a hypergraph with no bad edge pairs. Because the first step of Hybrid is to run MatchColoring and assign colors based on this algorithm, in theory is still enjoys the theoretical 2-approximation guarantee, and it can still make use of the explicit lower bound computed by this MatchColoring.

F. Experimental Results

Previous work has already shown that the LP relaxation can often be solved on real-world graphs and hypergraphs with tens of thousands of nodes and many large hyperedges within minutes (Amburg et al., 2020; 2022). The LP relaxation often even finds the optimal solution even using the simplest rounding techniques. Our improved LP approximation results therefore mainly serve as a way bring the best theoretical results closer to what has been observed in practice. On the other hand, our combinatorial algorithms provide a very practical new way to obtain approximate solutions with strong guarantees on a much larger scale than was ever possible previously. We implement our combinatorial algorithms in Julia. All experiments are run on a Mac laptop with 16GB of RAM. Source code and all data is publicly available on the Github repo `https://github.com/nveldt/ImprovedECC`.

Table 6. Statistics for five benchmark edge-colored hypergraphs from Amburg et al. (2020).

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	r	k
`Brain`	638	21180	2	2
`Cooking`	6714	39774	65	20
`DAWN`	2109	87104	22	10
`MAG-10`	80198	51889	25	10
`Walmart-Trips`	88837	65898	25	44

Experiments on previous benchmark hypergraphs Table 6 presents statistics for five benchmark edge-colored clustering hypergraphs first considered by Amburg et al. (2020). In Table 7, we show the performance of various algorithms in approximating ECC on these instances. The LP relaxation produces integral or near integral results on these datasets, so it is enough to apply a very simple rounding scheme (for each $v \in V$, assign it color $i^* = \text{argmin}_i\, x_v^i$) to obtain an optimal or near optimal solution to MINECC. PittColoring and MatchColoring exhibit a very straightforward tradeoff when compared with LP: they are far more scalable, at the expense of worse approximation guarantees. They nevertheless produce clusterings that are within a small factor of the LP lower bound (Approximation Factor in Table 7), and only take a fraction of a second.

The comparison between our combinatorial methods and MajorityVote is arguably more interesting. Although MajorityVote only has an r-approximation guarantee in theory, it often produces better clusterings than our combinatorial methods. However, on the `Walmart` dataset, PittColoring and MatchColoring produce better results. This may be due to the fact that this dataset has a much larger maximum hyperedge size, though it is not entirely clear. All of these combinatorial methods have the same asymptotic runtime of $O(\sum_e |e|)$, and very similar running times in practice. We have also reported results for running PittColoring and MatchColoring 100 times and taking the best result (Pitt+ and Match+ in Table 7). This is still very fast, and produces slightly better approximation factors. For each run, we randomize the order in which nodes are visited, which changes the order in which bad edge pairs are visited and therefore changes which hyperedges are deleted. PittColoring has additional randomization in how it chooses which edge to delete in a bad edge pair.

Experiments on the Trivago Hypergraph The `Trivago` hypergraph is derived from the 2019 ACM RecSys Challenge dataset (`https://recsys.acm.org/recsys19/challenge/`). It is closely related to the previous Trivago hypergraph without edge labels available at `https://www.cs.cornell.edu/~arb/data/trivago-clicks/`. The key difference is that in processing the data, we have kept track of the user location platform when parsing the website browsing data, which allows us to obtain edge country labels. We discard nodes that have labels that do not correspond to any of the hyperedge labels.

Table 2 in the main text displays the number of mistakes, edge satisfaction, a posteriori approximation ratio (for all but PittColoring, which does not compute an explicit lower bound), accuracy, and runtime of each combinatorial method, all averaged over 50 different trials. There is little to no variation between different runs. The a posteriori approximation guarantee for MatchColoring and Hybrid is obtained by taking the number of mistakes made by the method and dividing by the lower bound computed by MatchColoring. MajorityVote does not compute as good of a lower bound. In theory, this provides an r-approximation, because this method can be seen as the optimal solution to an alternative edge-colored clustering objective where the penalty at each hyperedge is the *number* of nodes in the hyperedge with a color that is different from the hyperedge's color. This is always within a factor r of the optimal MINECC objective. Using this fact, it is not hard to prove that the optimal MINECC objective will be lower bounded by

$$\frac{\sum_{e\in E}\sum_{v\in e}\mathbb{1}\left(Y_{\text{MV}}[v] \neq \ell(e)\right)}{r}, \tag{22}$$

where Y_{MV} is the MajorityVote clustering. Dividing the MINECC objective for MajorityVote by this lower bound on the optimal MINECC solution produces the approximation in Table 2. Although this is better than the theoretical 85-approximation, it is still very poor.

Table 7. Approximation factors (ratio between algorithm output and LP lower bound), edge satisfaction (percentage of edges satisfied by the clustering), and runtimes obtained by various algorithms on five benchmark edge-colored hypergraphs from Amburg et al. (2020). MajorityVote (MV) is deterministic. Our vertex cover algorithms are randomized, so we list the mean values and standard deviations over 50 runs. Pitt+ and Match+ correspond to running PittColoring and MatchColoring 100 times and taking the best clustering found.

	Ratio to LP lower bound					
Dataset	LP	MV	PittColoring	MatchColoring	Pitt+	Match+
`Brain`	1.0	1.01	1.07 $\pm$0.01	1.08 $\pm$0.01	1.06 $\pm$0.01	1.07 $\pm$0.01
`Cooking`	1.0	1.21	1.23 $\pm$0.01	1.23 $\pm$0.0	1.22 $\pm$0.0	1.22 $\pm$0.01
`DAWN`	1.0	1.09	1.57 $\pm$0.04	1.58 $\pm$0.03	1.54 $\pm$0.03	1.54 $\pm$0.03
`MAG-10`	1.0	1.18	1.39 $\pm$0.01	1.49 $\pm$0.0	1.37 $\pm$0.0	1.48 $\pm$0.0
`Walmart-Trips`	1.0	1.2	1.13 $\pm$0.0	1.18 $\pm$0.0	1.13 $\pm$0.0	1.17 $\pm$0.0

	Edge Satisfaction					
Dataset	LP	MV	PittColoring	MatchColoring	Pitt+	Match+
`Brain`	0.64	0.64	0.62 $\pm$0.0	0.62 $\pm$0.0	0.62 $\pm$0.0	0.62 $\pm$0.0
`Cooking`	0.2	0.03	0.01 $\pm$0.0	0.01 $\pm$0.0	0.02 $\pm$0.0	0.02 $\pm$0.0
`DAWN`	0.53	0.48	0.26 $\pm$0.02	0.25 $\pm$0.02	0.27 $\pm$0.01	0.27 $\pm$0.01
`MAG-10`	0.62	0.55	0.47 $\pm$0.0	0.44 $\pm$0.0	0.48 $\pm$0.0	0.44 $\pm$0.0
`Walmart-Trips`	0.24	0.09	0.14 $\pm$0.0	0.11 $\pm$0.0	0.14 $\pm$0.0	0.11 $\pm$0.0

	Runtime					
Dataset	LP	MV	PittColoring	MatchColoring	Pitt+	Match+
`Brain`	0.52	0.001	0.006 $\pm$0.028	0.002 $\pm$0.001	0.12 $\pm$0.006	0.028 $\pm$0.004
`Cooking`	127.01	0.002	0.01 $\pm$0.003	0.008 $\pm$0.007	0.525 $\pm$0.012	0.228 $\pm$0.011
`DAWN`	4.23	0.003	0.01 $\pm$0.004	0.005 $\pm$0.003	0.779 $\pm$0.038	0.221 $\pm$0.007
`MAG-10`	17.0	0.012	0.04 $\pm$0.006	0.04 $\pm$0.016	0.675 $\pm$0.037	0.414 $\pm$0.06
`Walmart-Trips`	321.09	0.07	0.053 $\pm$0.007	0.05 $\pm$0.009	1.545 $\pm$0.083	1.044 $\pm$0.067