Political districting to minimize cut edges

Hamidreza Validi[1] and Austin Buchanan[2]

[1]Computational and Applied Mathematics, Rice University,
hamidreza.validi@rice.edu
[2]Industrial Engineering & Management, Oklahoma State University,
buchanan@okstate.edu

April 21, 2021

Abstract

When constructing political districting plans, prominent criteria include population balance, contiguity, and compactness. The compactness of a districting plan, which is often judged by the "eyeball test," has been quantified in many ways, e.g., Length-Width, Polsby-Popper, and Moment-of-Inertia. This paper considers the number of cut edges, which has recently gained traction in the redistricting literature as a measure of compactness because it is simple and reasonably agrees with the eyeball test. We study the stylized problem of minimizing the number of cut edges, subject to constraints on population balance and contiguity. With the integer programming techniques proposed in this paper, all county-level instances in the USA (and some tract-level instances) can be solved to optimality. Our techniques easily extend to minimize *weighted* cut edges (e.g., to minimize district perimeter length) or to impose compactness *constraints*. All data, code, and results are on GitHub.

Keywords: political redistricting; contiguity; integer programming; branch-and-cut; cut edges; GerryChain; compactness; perimeter; exact algorithm

1 Introduction

Political redistricting is the process of partitioning a region (e.g., a state) into smaller pieces ("districts") for voting purposes. Typically, redistricting plans must satisfy criteria such as population balance, contiguity, and compactness. Population balance means that the districts should have (roughly) equal populations and captures the principle of "one person, one vote." Meanwhile, contiguity and compactness are meant to keep geographic locales together, as their inhabitants may share political interests that may become the subject of legislation. Contiguity and compactness are sometimes suggested as a means to combat the most egregious cases of gerrymandering (i.e., redistricting to benefit or disadvantage a particular group, often a political party or race), cf. Pennsylvania's "Twitter" plan [40].

While population balance and contiguity are relatively easy to codify, compactness has been more elusive, prompting dozens of alternative compactness scores proposed by political scientists, mathematicians, computer scientists, and operations researchers. This includes the Polsby-Popper score [36, 113, 114] and others that relate to perimeter [41, 112] or moment-of-inertia [69]. While every measure has its flaws [10, 11, 137], social scientists have observed that in practice many measures of compactness serve as reasonable proxies for the others [25, 103]. Still, no existing measure best mirrors the "eyeball test" [78]. Figure 1 shows optimally compact county-level districting plans for Oklahoma under the compactness measures of cut edges (left) and moment-of-inertia (right), under a 1% population deviation.

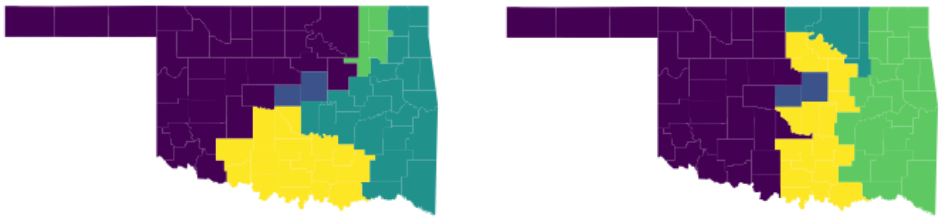

Figure 1: Optimally compact county-level districting plans for Oklahoma under the compactness measures of cut edges (left) and moment-of-inertia (right).

This paper considers the number of *cut edges*, which has recently gained traction in the redistricting literature as a measure of compactness, promoted in large part by members of the Metric Geometry and Gerrymandering Group [13, 34, 35, 36]. To define cut edges, the region of interest is represented as a graph $G = (V, E)$, where each vertex $i \in V$ represents a contiguous piece of the map (e.g., a county or census tract) and has an associated population p_i, and the graph's edges indicate adjacency on the map. The edges $\{i, j\} \in E$ that are *cut* are those whose endpoints i and j belong to different districts. Intuitively, the cut edges are those edges that would need to be snipped with a pair of scissors to break the graph into its districts.

Figure 2 provides two ways to partition the 4x4 grid graph into 4 contiguous districts having equal numbers of vertices. If compactness were measured via cut edges, then the *columns* plan, which cuts 12 edges, is the least compact plan possible. Meanwhile, the *squares* plan, which cuts 8 edges, is the most compact. Put differently, the columns plan *preserves* the fewest edges (12 edges), while the squares plan has 16 preserved, or *intact*, edges.

In this paper, we study a stylized redistricting problem in which the task is to minimize the number of cut edges, while ensuring that each of the k districts $D \subseteq V$ is contiguous (i.e., the induced subgraph $G[D]$ is connected) and has population $p(D) := \sum_{i \in D} p_i$ between L and U. In our experiments, we allow a 1% deviation ($\pm 0.5\%$) from the ideal population $p(V)/k$ by setting $L = \lceil 0.995 p(V)/k \rceil$ and $U = \lfloor 1.005 p(V)/k \rfloor$. This number is chosen to reasonably approximate what has been allowed for congressional redistricting in the USA [68] and has been a suggested threshold in a redistricting competition held by reformers in Ohio [5].

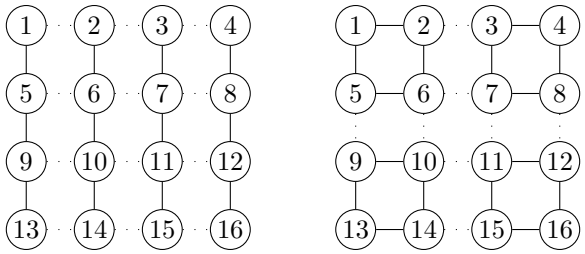

Figure 2: The *columns* plan cuts 12 edges, while the *squares* plan cuts 8 edges.

To solve this problem, we propose to use mixed integer programming (MIP) techniques. We start with two *base* MIP models—which we call Hess and Labeling—that are amended with various contiguity constraints, variable fixing procedures, valid inequalities, and extended formulations. The tractability of MIP models in practice can depend crucially on how these tools from the MIP arsenal are used. In both base models, we employ a binary variable y_e for each edge, which equals one when edge $e \in E$ is cut. The graph G has m edges and n vertices $\{1, 2, \ldots, n\}$. The set of k districts is given by $[k] := \{1, 2, \ldots, k\}$.

Labeling base model. The Labeling base model uses binary variables x_{ij} that equal one if vertex $i \in V$ is assigned to district $j \in [k]$. The essence of this model, particularly constraints (1b) and (1c), appears in many papers [13, 16, 47, 79, 125].

$$\min \sum_{e \in E} y_e \tag{1a}$$

$$x_{uj} - x_{vj} \leq y_e \qquad \forall e = \{u, v\} \in E,\ \forall j \in [k] \tag{1b}$$

$$\sum_{j \in [k]} x_{ij} = 1 \qquad \forall i \in V \tag{1c}$$

$$L \leq \sum_{i \in V} p_i x_{ij} \leq U \qquad \forall j \in [k] \tag{1d}$$

$$x \in \{0,1\}^{n \times k},\ y \in \{0,1\}^m. \tag{1e}$$

The objective (1a) minimizes the number of cut edges. Constraints (1b) indicate that edge $e = \{u, v\}$ is cut if vertex $u \in V$—but not $v \in V$—is assigned to district $j \in [k]$. Constraints (1c) ensure that each vertex $i \in V$ is assigned to one district. Constraints (1d) ensure that the population of each district is between L and U. As written, this model lacks contiguity constraints (discussed later).

We make a few notes about this model. First, the cut edge constraints (1b) only ensure that if u and v are assigned to different districts, then the associated cut edge variable y_e equals one. The converse is not imposed. That is, the model allows u and v to be assigned to the same district and still have y_e equaling one. Since our problem is of the minimization type (and since the objective coefficients

of y are positive), this is not a concern; if otherwise, one should add constraints of the form $x_{uj} + x_{vj} + y_e \leq 2$ to the model. A second note is also about the cut edge constraints (1b). For purposes of correctness, it suffices to impose one constraint for each edge: $x_{uj} - x_{vj} \leq y_e$. However, it can be desirable to impose two constraints for strength reasons: $x_{uj} - x_{vj} \leq y_e$ and $x_{vj} - x_{uj} \leq y_e$. Third, it suffices to define the y variables as nonnegative continuous variables, as they will take binary values in optimal solutions by the cut edge constraints and minimization objective. However, our preliminary experiments with this relaxation yielded no improvement; in fact, the Gurobi MIP solver converted them back to binary during presolve.

The Labeling model (1) has many undesirable properties [74]. First, it gives an extremely weak LP bound. In fact, under the mild assumption that $L \leq p(V)/k \leq U$ which is necessary for the LP to be feasible, the LP relaxation allows the solution $(\bar{x}, \bar{y})$ in which $\bar{x}_{ij} = 1/k$ and $\bar{y}_e = 0$, giving an LP bound of zero. This is confirmed empirically in Table 1, which reports results for 12 county-level instances that are solved with a naïve implementation of model (1) and the Gurobi MIP solver. (The test instances and computational setup will be detailed in Section 3.)

Table 1: Results for *naïve* implementations of models (1) and (2) under a 3600-second time-limit (TL). We report the number of branch-and-bound nodes visited (B&B), the LP bound (LP), the optimal MIP objective value (MIP) or the best lower and upper bound [LB,UB] at termination, and the MIP time in seconds (time). On right, we report whether the optimal solutions are contiguous.

			Labeling (1)				Hess (2)				opt.
state	n	k	B&B	LP	MIP	time	B&B	LP	MIP	time	cont.?
ME	16	2	1	0	8	0.07	1,480	0.27	8	2.32	no
NM	33	3	92	0	17	0.19	49,106	0.48	17	526.52	yes
ID	44	2	52	0	10	0.14	12,326	0.09	10	652.21	yes
WV	55	3	6,524	0	20	2.24	9,301	0.14	[1,20]	TL	no
LA	64	6	2,540,158	0	[36,45]	TL	3,708	0.77	[5,48]	TL	*
AL	67	7	1,586,159	0	[38,54]	TL	2,018	0.94	[14,55]	TL	*
AR	75	4	266,038	0	32	203.85	25	0.27	[3,45]	TL	no
OK	77	5	76,338	0	39	132.88	25	0.36	[4,49]	TL	no
MS	82	4	836,663	0	32	457.61	1	0.25	[2,62]	TL	no
NE	93	3	2,104	0	19	2.86	1	0.11	[1,30]	TL	yes
IA	99	4	529,067	0	33	716.04	7	0.18	[1,51]	TL	yes
KS	105	4	320,451	0	31	425.73	1	0.18	[1,53]	TL	no

* If given more time, these approaches would return "no".

Another drawback of this model is symmetry: if the vertices can be partitioned into k suitable districts $D_1, D_2, \ldots, D_k$, then permuting the district labels gives $k!$ different x-representations of this same districting plan. The point $(\bar{x}, \bar{y})$ that has $\bar{x}_{ij} = 1/k$ and $\bar{y}_e = 0$ will lie in their convex hull and thus be LP feasible, *even in the presence of contiguity constraints or other valid inequalities that act in the x-space.* Not surprisingly, the two states in Table 1 that have the most districts and arguably the "most" model symmetry are unsolved by Gurobi within a one-hour time-limit,

despite visiting more than one million branch-and-bound nodes. Model symmetry is generally known to cause trouble for LP-based branch-and-bound methods, which has led to a rich literature on symmetry handling [96, 111]. One approach that we will use is the extended formulation for partitioning orbitopes [46].

We also observe that contiguity does not come "for free" as some researchers have suggested. The cut edges objective function, which seeks compact solutions, tends to generate *nearly* contiguous solutions. However, in the absence of explicit contiguity constraints, the Labeling model gives non-contiguous solutions for 67% of the county-level instances, as the right-most column of Table 1 shows. We will see that the same phenomenon occurs on more granular tract-level instances.

Hess base model. The Hess base model starts with the binary variables x_{ij} of Hess et al. [69], which equal one when vertex $i \in V$ is assigned to (the district rooted at) vertex $j \in V$. This leads to the following model, whose essence, particularly constraints (2b)–(2e), appears in numerous papers [3, 104, 130, 131].

$$\min \sum_{e \in E} y_e \tag{2a}$$

$$x_{uj} - x_{vj} \leq y_e \qquad \forall e = \{u, v\} \in E,\ \forall j \in V \tag{2b}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \tag{2c}$$

$$L x_{jj} \leq \sum_{i \in V} p_i x_{ij} \leq U x_{jj} \qquad \forall j \in V \tag{2d}$$

$$\sum_{j \in V} x_{jj} = k \tag{2e}$$

$$x_{ij} \leq x_{jj} \qquad \forall i, j \in V \tag{2f}$$

$$x \in \{0,1\}^{n \times n}, y \in \{0,1\}^m. \tag{2g}$$

This model has many similarities to the Labeling model. It differs slightly because the x variables now allow each vertex i to be assigned to one of n different districts, and only k of these districts will be selected (2e). This requires a slight adjustment to the population balance constraints (2d). As written, this model lacks contiguity constraints. Later, we will review and experiment with some models from the literature [104, 124, 125, 131]. While the Hess model has more variables than the Labeling model, it does have one advantage: the variables x_{jj} act as anchor points for the districts, which can be helpful when writing contiguity constraints.

The Hess base model has many of the same deficiencies of the Labeling model—and more! For example, a valid k-partitioning $(D_1, D_2, \ldots, D_k)$ has $|D_1||D_2| \cdots |D_k|$ representations in the x variables, compared to $k!$ for the Labeling model. Not surprisingly, a naïve implementation of this model performs quite poorly, as Table 1 shows. The root LP bounds are all *less than one*, and only three of the instances are solved within a one-hour time-limit. Further, the MIP gaps at termination are awful. For example, the lower bound for West Virginia stays at *one* after one hour of computation, even though WV is one of the smaller instances.

One remedy for the model symmetry is to choose an ordering of the vertices $(v_1, v_2, \ldots, v_n)$ and impose $x_{ij} = 0$ whenever i comes before j in the ordering, like the asymmetric representatives model used for graph coloring problems [20]. This eliminates the model symmetry, but still leaves more than $n^2/2$ variables. Later, we will see how to choose orderings that lead to substantial size reductions in practice. On tract-level instances, more than 95% of the x variables can be fixed a priori.

Disclaimer. Political redistricting is, by its nature, a complex social matter, and we cannot decide the "best" redistricting plan with a computer alone. However, we hope that the procedures developed in this paper can nevertheless be useful for establishing the limits of what is possible in a redistricting plan, which can inform and assist socially aware redistricting efforts.

Outline. Section 2 reviews the literature on districting, compactness, contiguity constraints, and k-cut problems. Section 3 introduces our test instances and details our computational setup. Section 4 considers an extended formulation for the cut edge objective function that is stronger than (1a) and (2a). Section 5 covers heuristics. Sections 6 and 7 propose symmetry handling and variable fixing techniques for the Hess and Labeling models, respectively. Section 8 provides final computational experiments. Section 9 concludes the paper.

2 Background and Literature Review

The literature on political districting, gerrymandering, and graph partitioning is vast and cannot be covered in depth here. We refer interested readers to books and surveys written by operations researchers [56, 118, 121, 134], lawyers [68], political scientists [19, 60], and nonpartisan organizations [88].

As discussed previously, the essence of political districting is to partition a geographic region into a given number of districts that can be used for voting purposes. To abide by the "one-person, one-vote" principle, districts should have roughly the same number of people in them. In the USA, this is enforced quite closely, with all states' congressional districts drawn after the 2010 census differing in population by less than 1%. In fact, most states drew districts that differed in population by just one person(!). However, such a feat is impossible in some states that place high priority on preserving political subdivisions (e.g., counties, cities). For example, state law in Iowa dictates that counties should not be split between congressional districts. (The same is true in North Carolina, but doing so would result in a population deviation too large to abide by federal law, and federal law has supremacy.) There are other state and federal laws that apply to redistricting. For example, in the USA, Section 2 of the Voting Rights Act and the Equal Protection Clause of the 14th Amendment prohibit racial gerrymandering, although this is not as easy to quantify as population balance and is often litigated. Federal law does not require congressional districts to be contiguous, but most states require it. Even those states that do not require contiguity typically enact contiguous plans anyway.

Another *traditional redistricting principle* besides population balance and contiguity, is compactness. This *third criterion* [113] asks for districts to not be "ugly" in shape, preferring circular or square shapes over non-convex shapes with elongated tentacles. Indeed, in the landmark 1993 case *Shaw v. Reno*, the Supreme Court of the United States stated that redistricting "is one area in which appearances do matter." Besides the optics, the hope is that ensuring compactness will prevent the most egregious of gerrymanders and keep communities together. Over the years, many have tried to quantify what it means for a district to be compact. A popular technique is the Polsby-Popper score [113], which is defined as $4\pi A/P^2$, where A is the area of the district and P is its perimeter. The normalizing factor 4π ensures that the score is between zero and one, with a perfect score of one being given to a circle. This score, and others defined in terms of district perimeter, are known to suffer from the *coastline paradox* wherein borders do not have a well-defined length. Nefarious actors can exploit this and other implementation choices (e.g., map projection) to their advantage [10, 11] to hide from the Polsby-Popper score, and others like the Reock [116], Schwartzberg [122], and convex hull [103] scores.

An alternative compactness score proposed in the OR literature called moment-of-inertia [69] takes inspiration from physics and does not refer to area or perimeter. Letting d_{rj} denote the distance from a district's center/root r to its other vertices j, the moment-of-inertia of a district $D \subseteq V$ can be expressed as $\sum_{j \in D} p_j d_{rj}^2$, where p again represents population. The districting plan's moment-of-inertia is taken as the sum of the individual districts' scores. Since this is linear, it and related k-median objectives are convenient for use in OR models [32, 59, 71, 97, 129, 131].

In this paper, we consider the number of *cut edges*. While cut edges are frequently used in various graph partitioning and clustering applications [9, 15, 66], its use *in redistricting applications* is a relatively recent phenomenon [27, 39], promoted in large part by members of the Metric Geometry and Gerrymandering Group [13, 34, 35, 36]. The reason for using cut edges is threefold: (1) it is intuitive and easy to explain, making it suitable for non-experts to understand; (2) the data is transparent and easy to check, making it less prone to abuse; and (3) this simple score matches the "eyeball test" surprisingly well.

2.1 Districting complexity and methods

Practically any problem related to redistricting is NP-hard [4]. This is an immediate consequence of the population balance constraints, which can be used to express the NP-hard PARTITION problem. Moreover, redistricting problems remain hard even when working with *unit* populations. Dyer and Frieze [42] consider the problem of partitioning the vertices of a graph into connected subsets, each of size s. By reduction from PLANAR 3-DIMENSIONAL MATCHING, they show that this problem is NP-hard on planar bipartite graphs for every fixed $s \geq 3$. They also show that the problem remains hard when $s = n/2$. That is, partitioning a graph into two connected districts of equal size is NP-hard (i.e., our problem where $p = \mathbf{1}$, $k = 2$, and $L = U = n/2$). It should be noted that this second reduction of theirs does not generate planar instances.

Further, cut edge districting problems are NP-hard, even in the absence of population balance or contiguity constraints. In the minimum k-cut problem, the task is to partition the vertices into k (nonempty) subsets so as to minimize the weight of the edges between the subsets. This problem is generally NP-hard [58], but can be solved in time roughly $O(n^{k^2})$, which is polynomial when k is a fixed constant [58]. When one vertex is fixed in each subset, we get the multiterminal cut problem, which is NP-hard even for $k = 3$ in general graphs [31]. When the multiterminal cut problem is restricted to planar graphs, it remains NP-hard generally, but is polynomial for fixed constants k. Specifically, planar multiterminal cut is solvable in time $O(n^3 \log n)$ when $k = 3$, and generally in time $O((4k)^k n^{2k-1} \log n)$ [31]. This has been improved over the years [67, 135].

Given the hardness of districting, many techniques have been proposed. Heuristics in the literature run the gamut from greedy construction heuristics [80, 132] to local search [69, 81, 82, 83, 102, 131] to metaheuristics [6, 8, 17, 18, 63, 65, 90, 105, 119]. Other notable methods include generalizations of Voronoi diagrams [26, 87, 100, 117, 128], Markov chain Monte Carlo methods [2, 22, 34, 35, 36, 49], approximation algorithms [70], and fixed-parameter-tractable algorithms [27]. For more, see the surveys of Ricca et al. [118] and Goderbauer and Winandy [56].

Regarding MIP models, we have already seen the Labeling model and a Hess-style model for cut edges, which both use assignment variables of the form x_{ij} and (cut) edge variables of the form y_e. Chopra and Rao [23] perform a polyhedral study of the partition problem in the (x, y)-space of variables, where $y_e = 1$ if edge $e \in E$ is *preserved*. They consider two problem variants: one where the vertices are to be partitioned into *at most* k subsets, and another where they are to be partitioned into *at least* k subsets, neither imposing contiguity. For the former problem variant, they propose valid inequalities based on cycles, cliques, wheels, and bicycle wheels. Separation complexity is also studied. For the latter problem variant, valid inequalities are developed for spanning trees and supersets thereof. Later, Chopra and Rao [24] perform a polyhedral study in the (cut) edge space (with no assignment variables).

Relatedly, Grötschel and Wakabayashi [61] propose a cutting plane approach for the clique partitioning problem, which is the task of partitioning the vertices of a *complete* graph into clusters (cliques) so as to maximize the weight of the intra-cluster edges. In their implementation, Grötschel and Wakabayashi use the triangle inequalities and $[S, T]$-inequalities that they proposed in their separate paper [62] that act on the (preserved) edge variables. Oosten et al. [106] characterize all facet-defining inequalities for clique partitioning that have right-hand-side one or two. Deza et al. [37] develop clique-web facets for clique partitioning (over a complete graph) in which the number of subsets is restricted to be at most, or at least, k.

Ferreira et al. [47] conduct a polyhedral study of *vertex capacitated* graph partitioning. In terms of our notation, they consider the problem of partitioning the vertices of a graph into at most k subsets, with the requirement that each subset have weight at most U, so as to minimize the weight of the cut edges. Connectivity is not enforced. They introduce a strong extended formulation for the cut edges objective that we will discuss in Section 4. Much of their polyhedral study

is conducted in the space of edge variables y, often combining ideas from knapsack polyhedra (e.g., knapsack covers) and graph structures (e.g., cycles, stars, and trees). In a followup paper, Ferreira et al. [48] develop a branch-and-cut algorithm proposing to heuristically separate various valid inequalities from their former paper. The largest instance solved in their paper had 274 vertices and 469 edges, and used parameter values (in our terms) of $p = \mathbf{1}$, $k = 2$, and $U = \lceil n/2 \rceil$.

Labbé and Özsoy [84] conduct a polyhedral study for size-constrained graph partitioning problems. Here, the task is to partition the vertices of a graph into subsets, with the requirement that each subset has between L and U vertices in it (in our notation), so as to minimize (or maximize) the weight of the cut edges. Connectivity is not imposed. Because of the lower *and* upper size bounds, establishing the polytope's dimension is a nontrivial task. Under some conditions, they extend results about the $[S, T]$ inequalities of Grötschel and Wakabayashi [62]; the clique inequalities of Chopra and Rao [23]; and cycle (with ear) inequalities of Conforti et al. [28], Ferreira et al. [47], and Sørensen [126].

In a completely different MIP approach, some have proposed to use a set partitioning model, with a binary variable for each possible district. An early example is the approach taken by Garfinkel and Nemhauser [54] which first enumerates all satisfactory districts and then solves the resulting MIP. Applying their approach to county-level redistricting instances in the USA, they observe that instances with $n \leq 30$ are easy, while instances around $n = 50$ are difficult, with West Virginia failing to solve in one hour. Decades later, Mehrotra et al. [97] continue with a similar set partitioning model, but solve it with a branch-and-price approach, generating suitable districts on-the-fly. Using a compactness objective based on distances, they (approximately) solve a county-level MIP for South Carolina, and then manually adjust some assignments for better population balance. Mehrotra and Trick [98] propose a branch-and-price approach for clique partitioning and capacitated clustering problems, testing them on instances with up to 36 and 61 nodes, respectively. Observing the weak LP relaxation and symmetry inherent to the Labeling model, Johnson et al. [74] propose a set partitioning model and column generation approach for a min-cut clustering problem. Here the task is to partition the vertices into k subsets, with lower and upper size bounds on the subsets, so as to minimize the weight of the cut edges. Recently, Gurnee and Shmoys [64] develop a set partitioning and column generation approach for redistricting, using stochastic hierarchical partitioning to generate many districting plans.

2.2 Contiguity constraints

Two general approaches for imposing contiguity in a MIP include flow constraints and cut constraints. When done right, these approaches lead to integral formulations for spanning trees [91], and for other special cases of Steiner tree [57, 115]. This is convenient for problems in which the key decisions are which edges to select.

However, in many problems, like districting, the key decisions are at the vertex level. In this case, we turn to a (vertex) separator instead of an (edge) cut. Figure 3 gives an example in which the vertex set $C = \{3, 7, 11, 15\}$ *separates* vertices $a = 5$

and $b = 8$ in the graph. Thus, if a and b were to belong to the same (contiguous) district, then at least one vertex from C must join them. This leads to *a, b-separator inequalities* of the form $x_{aj} + x_{bj} \leq 1 + \sum_{c \in C} x_{cj}$ for the Labeling model, and $x_{ab} \leq \sum_{c \in C} x_{cb}$ for the Hess model. The former have been used for the connected maximum k-cut problem [72], and the latter have been used for other partitioning problems [104, 131]. It should be noted that similar inequalities were first used for selecting a *single* connected subset of vertices by several researchers [21, 50, 133]. We refer to models based on these inequalities as CUT models.

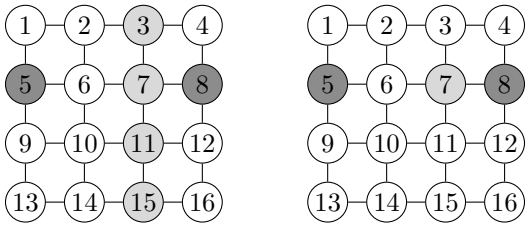

Figure 3: Illustration of a, b-separator (left) and length-U a, b-separator (right).

Validi et al. [131] observe that the a, b-separator inequalities can be strengthened by exploiting the population bound U. Specifically, C need not *fully* separate a from b; it only needs to disrupt all *short* paths between them. As long as all paths connecting a and b in $G - C$ have population greater than U, then the inequality $x_{aj} + x_{bj} \leq 1 + \sum_{c \in C} x_{cj}$ (or $x_{ab} \leq \sum_{c \in C} x_{cb}$) still applies. Following [120, 131], we call these inequalities length-U a, b-separator inequalities. Figure 3 gives an example showing that the a, b-separator $C = \{3, 7, 11, 15\}$ can be reduced to the length-U a, b-separator $C' = \{7\}$ when the graph is supposed to be split into four equal-size districts. In their experiments, Validi et al. [131] observe that these strengthenings improve the performance on county-level instances of districting, but not on the more granular tract-level instances where (minimal) a, b-separator inequalities already are (minimal) length-U a, b-separator inequalities. We refer to models based on length-U a, b-separator inequalities as LCUT models.

Definition 1. *Let $a, b \in V$ and $U \in \mathbb{R}$. A vertex subset $C \subseteq V \setminus \{a, b\}$ is called a length-U a, b-separator if $\text{dist}_{G-C,p}(a, b) > U$, where $\text{dist}_{G-C,p}(a, b)$ is the vertex-weighted distance from a to b in $G - C$ with respect to vertex weights p_i, $i \in V$.*

Since CUT and LCUT models generally have exponentially many constraints, it is important to understand the associated separation problems if one wants to use them. Assuming the graph is simple and planar, Validi et al. [131] show that fractional separation for the CUT model (with Hess variables) can be performed in time $O(n^2 \log n)$ by exploiting planar min-cut algorithms [77, 85]; when the point x^* to separate is integral, it takes time $O(n^2)$ using a procedure of Fischetti et al. [50] as a subroutine. For the LCUT model, fractional separation is NP-hard, but integer separation again takes time $O(n^2)$. Note that $O(n^2)$ is linear with respect to the number of Hess variables.

Now we turn to flow-based models, which are defined in terms of directed graphs. Accordingly, starting from $G = (V, E)$, replace each undirected edge $\{i, j\} \in E$ by two oppositely directed arcs (i, j) and (j, i) to get the directed graph $H = (V, A)$. The shorthands $\delta^-(i)$ and $\delta^+(i)$ refer to the subsets of arcs from H that point in and out of vertex i, respectively.

The most popular flow formulation is attributed to Shirabe [124, 125], see also [104, 131]. In it, k vertices are selected as district centers with the Hess variables ($x_{jj} = 1$); each district center generates flow that is then sent within the district, and one unit is consumed at its other vertices. Using a flow variable f^j_{uv} for each vertex (commodity) $j \in V$ and each arc $(u, v) \in A$, contiguity can be enforced with:

$$
\begin{aligned}
&& f^j(\delta^-(i)) - f^j(\delta^+(i)) &= x_{ij} && \forall i \in V \setminus \{j\},\ \forall j \in V && \text{(3a)}\\
\text{(Hess-SHIR)} && f^j(\delta^-(i)) &\le M x_{ij} && \forall i \in V \setminus \{j\},\ \forall j \in V && \text{(3b)}\\
&& f^j(\delta^-(j)) &= 0 && \forall j \in V && \text{(3c)}\\
&& f^j_{uv} &\ge 0 && \forall (u, v) \in A,\ \forall j \in V, && \text{(3d)}
\end{aligned}
$$

where $f^j(A') := \sum_{(u,v)\in A'} f^j_{uv}$ for edge subsets A'. Constraints (3a) enforce that if vertex i is assigned to another vertex j, then it should consume a unit of j's flow. The big-M constraints (3b) ensure that if i is not assigned to j, then no flow of type j can enter it. Traditionally, researchers have set $M = n - 1$, but we can use

$$M = \max_{D \subset V} \{|D| : p(D) \le U\} - 1, \tag{4}$$

which can be efficiently computed with a greedy algorithm. Constraints (3c) enforce that vertex j does not receive flow of its own type.

Next from the literature is a single-commodity flow (SCF) formulation that is written over the Labeling variables [72], cf. [101]. It has one flow variable f_{ij} associated with each arc (i, j). There are also binary variables r_{ij} indicating whether vertex $i \in V$ is the root of district $j \in [k]$. These root variables are helpful for imposing contiguity and also for symmetry handling. The constraints are given by:

$$
\begin{aligned}
&& \sum_{i \in V} r_{ij} &= 1 && \forall j \in [k] && \text{(5a)}\\
&& r_{ij} &\le x_{ij} && \forall i \in V,\ \forall j \in [k] && \text{(5b)}\\
\text{(Labeling-SCF)} && f(\delta^-(i)) - f(\delta^+(i)) &\ge 1 - M \sum_{j \in [k]} r_{ij} && \forall i \in V && \text{(5c)}\\
&& f_{ij} + f_{ji} &\le M(1 - y_e) && \forall e = \{i, j\} \in E && \text{(5d)}\\
&& f_{ij} &\ge 0 && \forall (i, j) \in A && \text{(5e)}\\
&& r_{ij} &\in \{0, 1\} && \forall i \in V,\ \forall j \in [k], && \text{(5f)}
\end{aligned}
$$

where Hojny et al. [72] mention setting $M = n - k + 1$. Because of our population balance constraints, we can use the stronger M defined in (4). Constraints (5a) force each district to have one root. Constraints (5b) state that vertex $i \in V$ cannot root

a district $j \in [k]$ to which it does not belong. Constraints (5c) force vertex i to consume flow if it is not a root. Constraints (5d) disallow flow across cut edges.

Based on the categorization given in Table 2, one can conceive of three other methods for imposing contiguity: Labeling-LCUT, Hess-SCF, and Labeling-SHIR. The Labeling-LCUT model is straightforward to envisage, and the other two models are detailed in Appendix A. We will test these models in our experiments.

	Contiguity Constraints			
Base model	CUT	LCUT	SCF	SHIR
Hess	[104, 131]	[131]	none known	[104, 129, 131]
Labeling	[72, 101]	none known	[72, 101]	none known

Table 2: Summarizing contiguity models used in previous work.

3 Test Instances and Computational Setup

As discussed in the introduction, the Hess and Labeling base models have several undesirable properties (e.g., model symmetry, weak LP relaxation, unnecessarily many variables). In the sections that follow, we propose a variety of techniques to improve their scalability (e.g., heuristics, variable-fixing procedures, symmetry handling, stronger extended formulations). As the techniques are introduced, we evaluate their effectiveness on real redistricting instances from the USA.

The data that we use originates from the 2010 Census[1] and was then processed by Daryl DeFord [33]. This includes the generation of the contiguity graphs $G = (V, E)$ which are not provided by the US Census Bureau directly and have to be constructed from the GIS shapefiles. Our experiments use the actual number of congressional districts (k) and populations ($p_i,\ i \in V$). For the population balance constraints, we impose a 1% deviation by setting $L = \lceil 0.995p(V)/k \rceil$ and $U = \lfloor 1.005p(V)/k \rfloor$, which is typical [5, 68, 131].

We consider all county-level instances and some tract-level instances. As shown in our previous work [131], not all of the 50 county-level instances are interesting from a computational perspective. Seven of them are trivial, with $k = 1$. Many others (like California or Texas) are overtly infeasible in the sense that they have a county whose population exceeds U. What remains are sixteen instances, four of which are infeasible when contiguity is imposed (and this can be shown computationally in a few seconds by the techniques in our previous paper [131]). This leaves twelve county-level instances ($16 \leq n \leq 105$) that we will use in our experiments. We also consider ten tract-level instances, specifically those states that are nontrivial ($k > 1$), not too big ($n \leq 750$), and connected. These instances are big enough to show the limits of our approach.

[1]The 2020 Census data that is used for redistricting was not available at the time of writing, and is not expected to be released until August 2021.

To illustrate our proposed techniques, we will often apply them to New Mexico. One reason is that New Mexico is relatively small, having $n = 33$ counties, $n = 499$ census tracts, and $k = 3$ congressional districts after the 2010 Census. Another reason is that New Mexico has a square-like shape that is convenient for our figures.

All experiments use a Dell Precision Tower 7000 Series (7810) machine running Windows 10 enterprise, x64, with Intel Xeon Processor E52630 v4 (10 cores, 2.2GHz, 3.1GHz Turbo, 2133MHz, 25MB, 85W) and 32 GB memory. Our MIP solver is Gurobi 9.1. Our code is written in Python to easily interface with GerryChain (for our heuristic) and GeoPandas (for drawing maps) and is available at `https://github.com/hamidrezavalidi/Political-Districting-to-Minimize-Cut-Edges`.

4 Extended Objective

Recall the cut edge constraints (1b) and (2b), which take the form $x_{uj} - x_{vj} \leq y_e$ for edges $e = \{u, v\} \in E$ and districts j. In this section, we illustrate the weakness of these constraints and discuss a class of valid inequalities that subsumes them. While there are exponentially many of these inequalities, we find a small extended formulation for them that is equally as strong. To illustrate, consider the example given in Figure 4 in which the task is to split the 4 vertices of the graph into 2 districts of equal size ($k = L = U = 2$ and $p = \mathbf{1}$).

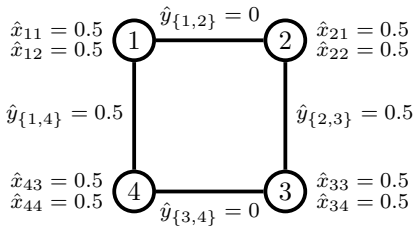

Figure 4: A fractional point $(\hat{x}, \hat{y})$ that is LP feasible for the Hess model (2).

The point $(\hat{x}, \hat{y})$ depicted in the figure is LP feasible for the Hess base model (2), and $\hat{x}$ also satisfies contiguity constraints. In fact, $\hat{x} = 0.5\bar{x} + 0.5\tilde{x}$ is a convex combination of x-feasible solutions $\bar{x}$ and $\tilde{x}$ in which $\bar{x}_{11} = \bar{x}_{21} = \bar{x}_{34} = \bar{x}_{44} = 1$ and $\tilde{x}_{12} = \tilde{x}_{22} = \tilde{x}_{33} = \tilde{x}_{43} = 1$. However, $\hat{y}$ does not reflect this. Specifically, observe that edge $\{1, 4\}$ is only *partially* cut ($\hat{y}_{\{1,4\}} = 1/2$), despite its endpoints 1 and 4 being assigned to completely different districts. In response, we could apply the following valid inequality that $(\hat{x}, \hat{y})$ violates.

$$(x_{43} + x_{44}) - (x_{13} + x_{14}) \leq y_{\{1,4\}}.$$

This inequality states that if vertex 4 is assigned to a vertex from the set $\{3, 4\}$ but vertex 1 is not, then the edge between them is cut. In fact, there is nothing special about the set $\{3, 4\}$, and the idea generalizes to arbitrary vertex subsets and edges.

Lemma 1 (cf. Ferreira et al. [47]). *If $S \subseteq V$ is a subset of vertices and $e = \{u, v\} \in E$ is an edge, then the following inequality is valid for the Hess base model* (2).

$$\sum_{j \in S} (x_{uj} - x_{vj}) \leq y_e \tag{6}$$

Proof. Suppose that $(\hat{x}, \hat{y})$ satisfies the Hess base model (2) and is thus binary. Without loss of generality, suppose that u is assigned to s ($\hat{x}_{us} = 1$) and that v is assigned to t ($\hat{x}_{vt} = 1$). In the first case, where $s \neq t$, we have $\hat{y}_e = 1$ by (2b), so

$$\sum_{j \in S} (\hat{x}_{uj} - \hat{x}_{vj}) \leq \sum_{j \in S} \hat{x}_{uj} \leq 1 = \hat{y}_e.$$

In the other case, where $s = t$, we have $\sum_{j \in S} (\hat{x}_{uj} - \hat{x}_{vj}) = 0 \leq \hat{y}_e$. □

Lemma 1 also applies to the Labeling model under the revised assumption that $S \subseteq [k]$, as observed by Ferreira et al. [47]. Also observe that the cut edge inequalities (1b) and (2b) are special cases of the strengthened version where $|S| = 1$.

Although these inequalities (6) can strengthen the LP relaxation, we do not use them. The reason is that there is a small extended formulation for them that is just as strong, and using it is easier than implementing a separation callback. The extended formulation is based on new variables z_e^j that are defined for every edge $e = \{u, v\} \in E$, with $u < v$, and district center $j \in V$. They indicate whether edge e is cut because u is assigned to j but v is not. For the Hess model, we use the following constraints instead of the cut edge constraints (2b).

$$x_{uj} - x_{vj} \leq z_e^j \qquad \forall e = \{u, v\} \in E,\ u < v, \forall j \in V \tag{7a}$$

$$y_e = \sum_{j \in V} z_e^j \qquad \forall e \in E \tag{7b}$$

$$z_e^j \geq 0 \qquad \forall e \in E,\ \forall j \in V. \tag{7c}$$

Essentially the same model applies in the Labeling case, by changing V to $[k]$. Similarly, the theorem below also applies to Labeling, as Ferreira et al. [47] show.

Theorem 1 (cf. Ferreira et al. [47]). *The strengthened cut edge inequalities* (6) *and the extended formulation* (7) *are equally strong. They are at least as strong as the cut edge constraints* (2b).

Proof. Denote by P the set of (x, y) that satisfy the strengthened cut edge inequalities (6). Denote by Q the set of (x, y, z) that satisfy the extended formulation (7). We show that $P = \text{proj}_{x,y} Q$.

($\subseteq$) Suppose that $(\hat{x}, \hat{y})$ belongs to P. We construct $\hat{z}$ such that $(\hat{x}, \hat{y}, \hat{z})$ belongs to Q. For each edge $e = \{u, v\} \in E$, with $u < v$, define

$$d_e := \max_{S \subseteq V} \left\{ \sum_{j \in S} (\hat{x}_{uj} - \hat{x}_{vj}) \right\},$$

and observe that a solution to this problem is given by

$$S_e := \{j \in V \mid \hat{x}_{uj} - \hat{x}_{vj} > 0\}.$$

Also note that $h_e := \hat{y}_e - d_e$ is nonnegative by assumption that $(\hat{x}, \hat{y})$ satisfies the strengthened cut edge inequalities (6). Finally, for each vertex $j \in V$, let

$$\hat{z}_e^j := \begin{cases} (\hat{x}_{uj} - \hat{x}_{vj}) + h_e/n & \text{if } j \in S_e, \\ h_e/n & \text{if } j \in V \setminus S_e. \end{cases}$$

By this definition, each $\hat{z}_e^j$ is nonnegative and thus satisfies constraints (7c).

Next, we show that $(\hat{x}, \hat{y}, \hat{z})$ satisfies constraints (7a). If $j \in S_e$, then we have

$$\hat{x}_{uj} - \hat{x}_{vj} \leq (\hat{x}_{uj} - \hat{x}_{vj}) + h_e/n = \hat{z}_e^j.$$

Otherwise, if $j \in V \setminus S_e$, then we have

$$\hat{x}_{uj} - \hat{x}_{vj} \leq 0 \leq h_e/n = \hat{z}_e^j.$$

Last, to show that constraints (7b) are satisfied, see that

$$\begin{aligned} \hat{y}_e = h_e + d_e &= \sum_{j \in V} (h_e/n) + \sum_{j \in S_e} (\hat{x}_{uj} - \hat{x}_{vj}) \\ &= \sum_{j \in V \setminus S_e} (h_e/n) + \sum_{j \in S_e} ((\hat{x}_{uj} - \hat{x}_{vj}) + h_e/n) \\ &= \sum_{j \in V \setminus S_e} \hat{z}_e^j + \sum_{j \in S_e} \hat{z}_e^j = \sum_{j \in V} \hat{z}_e^j. \end{aligned}$$

($\supseteq$) Suppose that $(\bar{x}, \bar{y}, \bar{z})$ belongs to Q. We show that $(\bar{x}, \bar{y})$ belongs to P, i.e., that $(\bar{x}, \bar{y})$ satisfies the strengthened cut edge inequalities (6). For this, consider an edge $e = \{u, v\} \in E$, with $u < v$, and vertex subset $S \subseteq V$, and see that

$$\sum_{j \in S} (\bar{x}_{uj} - \bar{x}_{vj}) \leq \sum_{j \in S} \bar{z}_e^j \leq \sum_{j \in V} \bar{z}_e^j = \bar{y}_e,$$

where the first inequality holds by constraints (7a), the second inequality holds by constraints (7c), and the equality holds by constraints (7b). □

Later, in Subsection 8.1, we will see that this extended objective can be quite helpful in practice, leading to substantial improvements in the root LP bound and in the number of branch-and-bound nodes.

5 Heuristic

In preliminary experiments, we observed that the MIP solver sometimes had trouble finding a good initial solution. We also observed that, as soon as a good solution was

found, the MIP solver could often prove optimality quite quickly. With this in mind, we sought to use a heuristic to warm-start the solve process. Rather than reinvent the wheel, we use an existing codebase called GerryChain v0.2.12 [99]. GerryChain is based on a Markov chain Monte Carlo (MCMC) framework, repeatedly moving from one feasible solution to a (randomly chosen) neighboring feasible solution, much like local search. A difference is that GerryChain was primarily intended to generate large collections of redistricting plans, with no particular objective function in mind. In the intended use, a proposed (or enacted) plan can be compared to the resulting "distribution" of plans to see whether it is a (gerrymandered) outlier. Nevertheless, we find that GerryChain works reasonably well as a heuristic for minimizing the number of cut edges.

GerryChain employs two search neighborhoods, referred to as flip and recombination. In flip, a vertex on the boundary between two districts can be moved to the other district. Thus, only small changes can be made with the flip neighborhood. Meanwhile, recombination allows for much larger changes [35]. In it, two (or more) adjacent districts are temporarily merged into one district, a random spanning tree is constructed within them, and some edges are removed from the spanning tree to split it into the appropriate number of districts. This process is repeated if the recombination move gives an infeasible solution. In our experiments, we follow the suggested practice of running GerryChain for 10,000 iterations, as this is empirically when the Markov chain appears to reach steady state [34, 35].

Figure 5 depicts the best solutions found for New Mexico within 10,000 iterations. The districting plan on the left gives a county-level solution that cuts 17 edges, which turns out to be optimal. On the right is a tract-level solution that cuts 46 edges, which is reasonably close to the optimal objective 43.

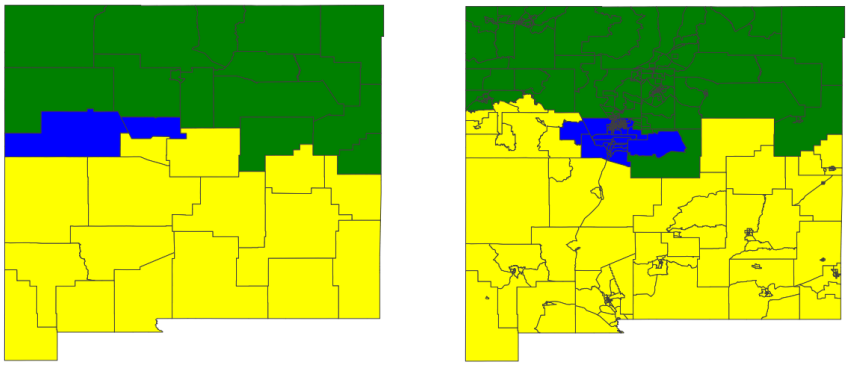

Figure 5: Heuristic county-level and tract-level solutions for New Mexico.

In Table 3, we provide more details on our experience with GerryChain. Specifically, we report the heuristic's objective value as the number of iterations increases from 100 to 1,000 to 10,000. For comparison, we also report the optimal objective value. Lastly, we report the time used by the heuristic.

Table 3: Experimental results using GerryChain as a heuristic. Results are provided for 100 and 1,000 and 10,000 iterations on county-level and tract-level instances.

			objective value				heuristic time		
state	n	k	100	1,000	10,000	opt	100	1,000	10,000
ME	16	2	-	-	-	16	-	-	-
NM	33	3	17	17	17	17	1.38	14.89	148.00
ID	44	2	10	10	10	10	5.14	39.50	404.51
WV	55	3	23	23	23	23	6.16	79.31	831.17
LA	64	6	-	-	-	49	-	-	-
AL	67	7	55	58	55	55	5.30	20.69	281.53
AR	75	4	33	33	33	33	3.20	41.69	402.48
OK	77	5	46	41	40	40	4.30	47.27	415.23
MS	82	4	34	34	34	34	5.39	46.46	485.37
NE	93	3	19	19	19	19	3.02	26.66	279.04
IA	99	4	36	33	33	33	3.81	63.70	577.47
KS	105	4	40	32	32	32	16.45	124.96	1,294.27
NH	295	2	30	27	26	26	13.31	141.68	1,319.08
ID	298	2	17	17	17	17	35.70	286.54	3,249.60
ME	358	2	22	21	20	20	18.86	125.51	1,471.17
WV	484	3	59	48	44	43	26.80	232.34	2,310.90
NM	499	3	48	48	46	43	28.70	245.92	2,481.29
NE	532	3	55	51	47	44	20.92	190.51	1,802.43
UT	588	4	110	107	97	-	21.56	221.17	2,310.29
MS	664	4	82	72	69	-	20.86	194.32	2,035.84
AR	686	4	88	85	79	-	21.55	197.66	1,903.86
NV	687	4	105	95	89	-	26.67	224.15	2,195.91

We observe that the objective values tend to improve as the number of iterations increases, although this is not guaranteed because of the random nature of GerryChain (e.g., see AL). We see that GerryChain is able to find optimal solutions for most of the county-level instances within 10,000 iterations. Exceptions are Maine and Louisiana. In fact, in our experiments, GerryChain stalls on Maine and does not terminate. Investigating this issue, we find that Maine has only one feasible districting plan at the county level, leaving no room for flip and recombination moves. Meanwhile, on Louisiana, GerryChain was simply unable to find a feasible starting solution, even though one exists. Such behavior is bound to happen in some cases given that GerryChain is not an exact method. This provides additional motivation for the exact procedures proposed in this paper. On county-level instances, the running times are reasonable, taking seconds when the number of iterations is 100, and taking minutes when the number of iterations is 10,000. If desired, these times could be improved by implementing GerryChain in a different programming language (e.g., Julia[2] or C++); however, this is outside our scope.

[2]While we were working on this paper, MGGG began implementing GerryChain in Julia [127].

On the tract-level instances, the performance of GerryChain is similar, but slightly worse. On several states, GerryChain runs for 10,000 iterations without arriving at an optimal solution (e.g., for WV, NM, NE), although they are close. Better solutions could be found with more iterations, but we chose not to given that GerryChain was already taking roughly 30 minutes to complete 10,000 iterations.

6 Symmetry and Fixing for Hess

In this section, we seek to improve the performance of the Hess model, primarily by reducing the size of the MIP by safely fixing variables to zero. First, we propose *diagonal-fixing*, which works as a symmetry-breaking technique and also as a way to cut its variables nearly in half. Next, we propose *L-fixing*, which exploits the population lower bound L to fix some center variables x_{jj} to zero, which also fixes the associated variables x_{ij} for $i \in V$ in the process. Then, we propose *U-fixing*, which exploits the population upper bound U to fix some variables x_{ij} to zero when vertices i and j are "far apart" from each other and impossible to belong to the same district. Last, we propose *Z-fixing*, in which we safely fix some of the variables z_e^j to zero. While these procedures are primarily intended for the Hess base model (which needs a size reduction the most), we will later see that they can be extended to the Labeling model.

These variable fixing procedures can be quite powerful, as Table 5 illustrates for New Mexico (NM). Since NM has 33 counties, the Hess base model uses $(33)^2 = 1,089$ variables of the type x_{ij}. Of them, 528 are fixed to zero by diagonal-fixing, 406 by L-fixing, and 28 by U-fixing, as reported in Table 4. In total, the number of x variables drops from $1,089$ to 127, a reduction of 88%. Additionally, Bernalillo County and Doña Ana County, which cannot be assigned to other counties, must root their own districts ($x_{jj} = 1$) by the assignment constraints (2c). Further, 88% of the z variables can be fixed to zero. The percentage of variables that are fixed by our techniques increases to 95% or more on the tract-level instances.

Table 4: Experiments with Hess variable fixing and time limit (TL) of 60 seconds. We report the size of set B obtained via model (12), and MIP solve time in seconds. Next are the number of x variables fixed via diagonal-fixing (DFix), L-fixing (LFix), and U-fixing (UFix), followed by the total percentage of x variables fixed (%X) rounded to the nearest percent. Last is the percentage of z variables fixed (%Z).

			model (12)		How many variables are fixed?				
state	n	k	$\|B\|$	time	DFix	LFix	UFix	%X	%Z
ME	16	2	13	0.03	120	91	0	82	83
NM	33	3	28	0.17	528	406	28	88	88
ID	44	2	41	0.05	946	861	0	93	93
WV	55	3	48	3.62	1,485	1,176	20	89	89
LA	64	6	53	30.68	2,016	1,431	58	86	85
AL	67	7	52	TL	2,211	1,378	110	82	83
AR	75	4	64	24.01	2,775	2,080	28	87	87
OK	77	5	64	20.83	2,926	2,080	101	86	86
MS	82	4	69	TL	3,321	2,415	20	86	86
NE	93	3	86	2.56	4,278	3,741	5	93	93
IA	99	4	85	TL	4,851	3,655	52	87	87
KS	105	4	95	12.55	5,460	4,560	11	91	91
NH	295	2	283	7.61	43,365	40,186	0	96	96
ID	298	2	291	2.48	44,253	42,486	138	98	98
ME	358	2	349	3.19	63,903	61,075	0	98	98
WV	484	3	467	TL	116,886	109,278	1,195	97	97
NM	499	3	485	TL	124,251	117,855	184	97	97
NE	532	3	513	TL	141,246	131,841	375	97	97
UT	588	4	556	TL	172,578	154,846	918	95	95
MS	664	4	634	TL	220,116	201,295	1,456	96	96
AR	686	4	653	TL	234,955	213,531	1,982	96	96
NV	687	4	655	TL	235,641	214,840	707	96	96

Table 5: Variable fixings for New Mexico at the county level. Here, D, L, and U denote diagonal-fixing, L-fixing, and U-fixing, respectively.

County	Population	#	19	20	27	23	15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	16	17	18	21	22	24	25	26	28	29	30	31	32	33
Bernalillo	662,564	19		D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Doña Ana	209,233	20	U		D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Santa Fe	144,170	27	U			D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Rio Arriba	40,246	23	U				D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Cibola	27,213	15						D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Harding	695	1	U					L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Sierra	11,988	2	U					L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Lea	64,727	3	U					L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Guadalupe	4,687	4						L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Torrance	16,383	5						L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Grant	29,514	6	U					L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Otero	63,797	7	U					L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
San Juan	130,044	8	U					L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Roosevelt	19,846	9	U					L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Curry	48,376	10	U					L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Taos	32,937	11	U					L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Hidalgo	4,894	12	U					L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Eddy	53,829	13	U					L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
De Baca	2,022	14						L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Quay	9,041	16	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D	D
Colfax	13,750	17	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D	D
Los Alamos	17,950	18	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D	D
Chaves	65,645	21	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D	D
Valencia	76,569	22	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D	D
San Miguel	29,393	24	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D	D
Catron	3,725	25	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D	D
Sandoval	131,561	26	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D	D
Socorro	17,866	28	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D	D
Lincoln	20,497	29	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D	D
McKinley	71,492	30	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D	D
Luna	25,095	31	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D	D
Mora	4,881	32	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	D
Union	4,549	33	U					L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L

6.1 Symmetry handling via diagonal-fixing

As discussed in the introduction, the Hess base model has considerable model symmetry, allowing for $|D_1||D_2|\cdots|D_k|$ different representations of the same districting plan $(D_1, D_2, \ldots, D_k)$. This symmetry can be broken by picking an ordering of the vertices $(v_1, v_2, \ldots, v_n)$ and then fixing $x_{ij} = 0$ whenever vertex i comes before vertex j in the ordering [20]. In other words, fix

$$x_{ij} = 0 \text{ if } \operatorname{pos}(i) < \operatorname{pos}(j), \tag{8}$$

where $\operatorname{pos}(i)$ is the position of vertex i in the ordering $(v_1, v_2, \ldots, v_n)$, i.e., if $i = v_q$ then $\operatorname{pos}(i) = q$. This forces a canonical "center" for each district: its earliest vertex in the ordering. We call this diagonal-fixing because all entries of the matrix x that lie above the main diagonal are fixed to zero (after the rows and columns of x have been rearranged based on the ordering). Thus, nearly half of the Hess variables will be fixed, more specifically $(n^2 - n)/2$, see Table 5.

In the next subsection, we will see that the ordering can dramatically impact the number of variables that can be fixed through other means (e.g., L-fixing). So, we will seek to choose the ordering intelligently.

6.2 L-fixing

After diagonal-fixing, only certain vertices can be assigned to vertex $j \in V$, specifically only those vertices i that occur later in the ordering:

$$V_j = \{i \in V \mid \operatorname{pos}(i) \geq \operatorname{pos}(j)\}. \tag{9}$$

Often, this set V_j is so small (in population), that a feasible district cannot be built from it. In this case where $p(V_j) < L$, we can fix $x_{jj} = 0$. Further, by the coupling inequalities $x_{ij} \leq x_{jj}$, we can fix $x_{ij} = 0$ for all $i \in V_j$. If we would like to maximize the number of variables fixed in this way, then the ordering $(v_1, v_2, \ldots, v_n)$ should be constructed by sorting the vertices from largest to smallest population.

However, if we exploit the fact that districts should be contiguous, we can do more. That is, not all vertices of V_j can really be assigned to j; for example, vertices from other (connected) components of $G[V_j]$ cannot. This allows us to refine V_j to the subset of vertices S_j that can be reached by a path from j in $G[V_j]$:

$$S_j = \{i \in V_j \mid \text{there is an } i,j\text{-path in } G[V_j]\}. \tag{10}$$

Like before, if $p(S_j) < L$, then we can fix $x_{jj} = 0$, or more generally $x_{ij} = 0$ for all $i \in V_j$. We refer to this as L-fixing. As Table 5 illustrates, L-fixing can be quite powerful, provided that a good ordering is used. Fortunately, we have Lemma 2 to aid us in finding a good ordering.

Lemma 2. *Suppose that B is a subset of vertices and that every component of $G[B]$ has population less than L. If B is placed at the back of the ordering $(v_1, v_2, \ldots, v_n)$, then every vertex j from B will be L-fixed.*

Proof. Let $G[B_j]$ be the component of $G[B]$ that contains j. Observe that $p(B_j) < L$ by assumption that components of $G[B]$ have population less than L. Then, since $S_j \subseteq B_j$, we have $p(S_j) \leq p(B_j) < L$, so j is L-fixed. □

To illustrate Lemma 2, consider the 5x5 grid instance depicted in Figure 6. Here the task is to split the graph into 5 districts of equal size. On left, we show a maximum independent set B with 13 vertices, which would lead to 13 L-fixings of the form $x_{jj} = 0$, or $\binom{13+1}{2} = 91$ L-fixings of the form $x_{ij} = 0$. Note that planar graphs always admit an independent set of size at least $n/4$ by the four color theorem [7]. So, planar districting instances (that satisfy the mild condition $p_i < L$ for all $i \in V$) allow for at least $n/4$ L-fixings of the form $x_{jj} = 0$.

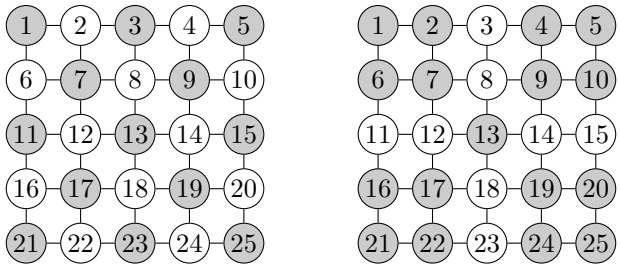

Figure 6: Vertices B that could be L-fixed if at back of ordering, when $L = 5$.

However, we can often do better, as the right side of Figure 6 illustrates. It shows a maximum cardinality subset of vertices B that satisfies the conditions of Lemma 2. If these vertices are placed at the end of the ordering, this results in 17 L-fixings of the form $x_{jj} = 0$, or $\binom{17+1}{2} = 153$ L-fixings of the form $x_{ij} = 0$. It turns out that putting a maximum such B at the back of the ordering gives a maximum number of L-fixings, not just for Figure 6, but generally, as Theorem 2 shows.

Theorem 2. *If a solution B to the following problem is placed at the back of the ordering, this maximizes the number of L-fixings.*

$$\max_{B \subseteq V} \{|B| : \textit{every component of } G[B] \textit{ has population less than } L\}. \tag{11}$$

Proof. Suppose that B solves problem (11) and let $(v'_1, v'_2, \ldots, v'_n)$ be an ordering that places B at the back. By Lemma 2, this gives $|B|$ L-fixings of the type $x_{jj} = 0$. Now, consider an arbitrary ordering $(v_1, v_2, \ldots, v_n)$, define V_j and S_j accordingly, and denote by $F = \{j \in V \mid p(S_j) < L\}$ as the subset of L-fixed vertices. See that F is a feasible solution to problem (11). (If F were infeasible, this means that $G[F]$ has a component $G[V']$ with $p(V') \geq L$ and its earliest vertex $v \in V'$ in the ordering has $V' \subseteq S_v$, giving the contradiction $L \leq p(V') \leq p(S_v) < L$.) So, since F is feasible for problem (11) while B is feasible and *maximum*, we have $|F| \leq |B|$. Thus, the ordering $(v'_1, v'_2, \ldots, v'_n)$ with B at back has at least as many L-fixings as the arbitrary ordering $(v_1, v_2, \ldots, v_n)$. □

We remark that the converse of Theorem 2 does not hold. For example, for the path graph 1-2-3-4-5 and $L = 2$, the ordering $(4, 5, 2, 1, 3)$ maximizes the number of L-fixings but does not place the unique optimal solution $B = \{1, 3, 5\}$ to problem (11) at the back.

6.3 An IP to solve the max B problem (11)

Since we would like to maximize the number of L-fixings, Theorem 2 tells us that we should seek orderings that place a solution B to problem (11) at the back. While problem (11) is NP-hard even on planar graphs [89, Corollary 5], it still may be worth solving if this leads to a commensurate speedup for the cut edge districting problem. Related interdiction problems have been studied in the literature, e.g., in which the task is to delete a minimum (weight) subset of vertices so that each (of the at most k) remaining component(s) has at most some number of vertices [12, 14, 16, 29, 52, 53, 86, 107, 123, 136].

To solve the maximum B problem (11), we "just MIP it" [51]. For every vertex $i \in V$, introduce a binary variable b_i that equals one if vertex i is selected in B. We also have binary variables x_{ij} that equal one when vertex $i \in V$ is assigned to "bin" $j \in [q]$. Later, we will decide how many bins q are necessary.

$$\max \sum_{i \in V} b_i \tag{12a}$$

$$\sum_{j \in [q]} x_{ij} = b_i \qquad \forall i \in V \tag{12b}$$

$$\sum_{i \in V} p_i x_{ij} \leq L - 1 \qquad \forall j \in [q] \tag{12c}$$

$$x_{uj} + b_v \leq 1 + x_{vj} \qquad \forall \{u, v\} \in E,\ \forall j \in [q] \tag{12d}$$

$$x \in \{0,1\}^{n \times q},\ b \in \{0,1\}^n. \tag{12e}$$

The objective (12a) maximizes the number of vertices in B. Constraints (12b) ensure that vertex i is selected in B if and only if it is assigned to one of the bins $j \in [q]$. Each of these bins has population less than L by constraints (12c). By constraints (12d), the bins do not touch each other. More specifically, they impose that if vertex u is assigned to bin j and its neighbor v is selected in B, then v must also be assigned to bin j. Thus, every component of $G[B]$ will have population less than L, as desired. In our implementation, we impose constraints (12d) for both orientations of edge $\{u, v\} \in E$. Observe that this model has $O(qn)$ variables, constraints, and nonzeros for simple planar graphs, but that projecting out the b_i variables would increase the number of nonzeros in constraints (12d) to $\Omega(q^2 n)$.

Below, we prove that $q = 2k$ bins suffice for our instances. This holds, for example, when $k \leq 99$, $L \geq 0.995$, and $\bar{p} \geq 39,800$, where $\bar{p} = p(V)/k$ is the ideal district population. For reference, California, the most populous state, had $k = 53$ congressional districts after the 2010 Census. Meanwhile, the state with the smallest ideal district population $\bar{p}$ was Rhode Island, with $\bar{p} = 527,623.50$.

Proposition 1. *For our instances, $q = 2k$ bins suffice. Generally, this holds if $k \leq 99$ and ideal district population $\bar{p} = p(V)/k$ satisfies $L \geq 0.995\bar{p}$ and $\bar{p} \geq 39,800$.*

Proof. Consider a feasible solution to problem (11) given by $B \subseteq V$ and let b^* be its characteristic vector. Let $G[B_1]$, $G[B_2]$, $\ldots$, $G[B_t]$ be the components of $G[B]$. By feasibility of B, each component $G[B_j]$ has population less than L. Pack these vertex subsets $B_1, B_2, \ldots, B_t$ into bins of capacity $L-1$ using the first-fit algorithm, giving new subsets $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_q$. This bin packing admits an associated x-representation x^* for which (x^*, b^*) satisfies the constraints of model (12) and has objective $|B|$.

It remains to show that $q \leq 2k$. Consider arbitrary bins $\mathcal{B}_i$ and $\mathcal{B}_j$, with $i < j$, from our bin packing. By the first-fit algorithm, greater than half of their combined capacity $2(L-1)$ is used, since otherwise it would have instead placed the items from $\mathcal{B}_j$ into $\mathcal{B}_i$ (or another earlier bin). Thus, letting $\mathcal{B}_{q+1} \equiv \mathcal{B}_1$, we have

$$k\bar{p} = p(V) \geq \sum_{i=1}^{q} p(\mathcal{B}_i) = \frac{1}{2}\sum_{i=1}^{q} \left(p(\mathcal{B}_i) + p(\mathcal{B}_{i+1})\right) > \frac{1}{2}\sum_{i=1}^{q}(L-1) = \frac{q(L-1)}{2},$$

which implies that $q < 2k\bar{p}/(L-1)$. Then, by our assumption that $L \geq 0.995\bar{p}$,

$$q < \frac{2k\bar{p}}{L-1} \leq \frac{2k\bar{p}}{0.995\bar{p}-1} = \frac{2k(0.995\bar{p}-1+0.005\bar{p}+1)}{0.995\bar{p}-1} = 2k + \frac{2k(0.005\bar{p}+1)}{0.995\bar{p}-1}.$$

Then, $q < 2k+1$ holds, because the right-most term above is at most one by

$$\frac{2k(0.005\bar{p}+1)}{0.995\bar{p}-1} \leq 198\left(\frac{0.005\bar{p}+1}{0.995\bar{p}-1}\right) \leq 198\left(\frac{1}{198}\right) = 1,$$

where the inequalities hold by $k \leq 99$ and $\bar{p} \geq 39,800$, respectively. So, $q \leq 2k$. □

The $2k$ bin bound from Proposition 1 is best-possible (under our assumptions). For example, consider a star graph $K_{1,4}$ whose hub vertex 1 has $p_1 = 1$ and each leaf $l \in \{2,3,4,5\}$ has $p_l = M$ sufficiently large, giving $p(V) = 4M+1$. Let $k = 2$, $L = 2M$, and $U = 2M+1$. The unique solution to the maximum B problem (11) is $B = \{2,3,4,5\}$, which requires $4 = 2k$ bins of capacity $L-1 = 2M-1$.

While setting $q = 2k$ is "safe" for solving the maximum B problem with model (12), it might be preferable in practice to use a smaller "unsafe" value, like $q = k$. This may compromise exactness (as the star example shows), but it yields a smaller MIP that can be more easily handled. In Table 15 of Appendix B, we provide results for both $q = k$ and $q = 2k$, under one-minute and one-hour time-limits. As the results show, to get a practically large set B, we can set $q = k$ and use a one-minute time-limit. This is inexact, e.g., 291 versus 292 for ID at the tract level, but this is fine as we only use the resulting set B for L-fixing purposes.

Another way we help the MIP solver find good solutions to model (12) is to give it a *partial* warm start solution. The idea is as follows. Suppose we have an initial districting plan $D_1, D_2, \ldots, D_k$ obtained via, say, GerryChain. This plan gives a feasible solution x^* to the Labeling model. In it, each vertex $i \in V$ will be assigned

to one district $j \in [k]$. In our partial warm start, we suggest $x_{it} = 0$ for all $t \neq j$. After this suggestion, we have $x_{ij} = b_i$ and model (12) reduces to the following.

$$\max \sum_{i \in V} b_i \tag{13a}$$

$$\sum_{i \in D_j} p_i b_i \leq L - 1 \qquad \forall j \in [k] \tag{13b}$$

$$b_u + b_v \leq 1 \qquad \forall \{u, v\} \in \delta(D_1, D_2, \ldots, D_k) \tag{13c}$$

$$b \in \{0, 1\}^n. \tag{13d}$$

The objective (13a) maximizes the number of vertices selected in B. Constraints (13b) ensure that, from each district D_j, we select a subset of vertices whose population is strictly less than L. Constraints (13c) ensure that for each cut edge $\{u, v\}$ we cannot select both of its endpoints; in this way, the vertices D'_j selected from within district D_j do not touch the other district subsets D'_t, $t \neq j$. This model (13) is quite small, easy to solve in practice, and gives good solutions. In fact, problem (13) is fixed-parameter tractable (fpt) with respect to the dual parameter $n - |B|$ by a simple, bounded search tree algorithm; see Downey and Fellows [38] or Cygan et al. [30] for more about this generic algorithmic technique. A possible algorithm for problem (13) is as follows.

1. if all cut edges $\{u, v\}$ have at least one of b_u and b_v fixed to zero, then solve the resulting problem using a greedy (exact) algorithm;
2. otherwise, pick a cut edge $\{u, v\}$ for which both b_u and b_v are "free." Create two subproblems. In first subproblem, fix $b_u = 0$. In second subproblem, fix $b_v = 0$. Solve the subproblems recursively by calling step 1.

Suppose that the search tree is explored in a breadth-first manner. In this way, if there is a solution B of size $|B| = n - t$, then it will be found on level t (or higher up in the tree) since one b variable is fixed to zero in each node of the search tree. So, the algorithm can stop at level t and thus visits at most $2^{t+1} - 1$ nodes. The greedy algorithm in step 1 runs in time $O(n \log n)$. So, the total time is bounded by $O(2^t n \log n)$. In our experiments, the value of t is typically small, often around ten or twenty. We think this partially explains why the (partial) warm start that we give for model (12) is so helpful.

When we apply model (12) to New Mexico at the county-level, it identifies a solution B consisting of all but five counties, $V \setminus B = \{15, 19, 20, 23, 27\}$. This is depicted in Figure 7. Interestingly, the set of "interdicted" counties (in gray fill) not only splits the graph into pieces (e.g., by 15, 19, 23, 27), it also simply removes a high-population vertex (20). By spending less than one second to solve model (12), we can fix a total of $\binom{28+1}{2} = 406$ variables from the Hess model, see Table 4.

As seen in Table 4, many of the county-level instances of problem (11) are easily solved with model (12) and allow us to fix more than 80% of the center variables x_{jj}. On the tract-level instances, we spend at most one minute on model (12) and fix over 95% of the center variables, as the $|B|$ column of Table 4 shows. The right side of Figure 7 depicts the solution obtained by model (12) for New Mexico.

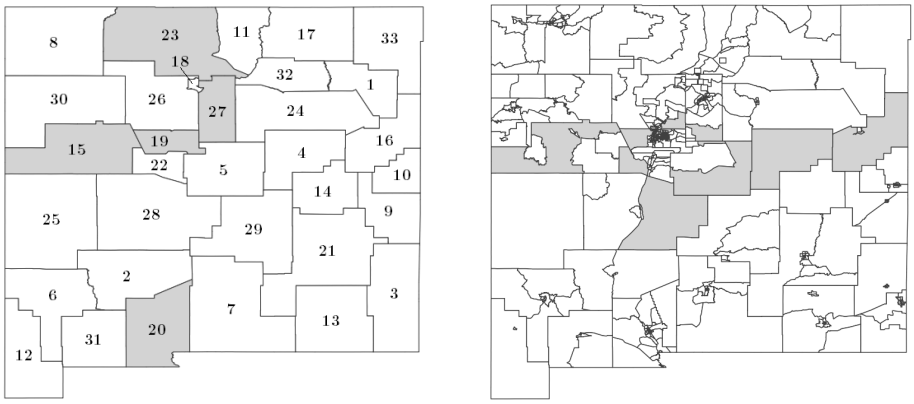

Figure 7: Illustration of L-fixing for New Mexico. On left, an optimal county-level solution (in white) for problem (11) obtained by solving model (12). On right, the best tract-level solution to problem (11) obtained in one minute with model (12).

6.4 U-fixing

Previously, we used the population lower bound L to safely fix x variables. In this subsection, we exploit the population upper bound U. The main insight is that if vertices i and j are sufficiently "far apart" then they cannot belong to the same district, in which case we can fix $x_{ij} = 0$. Specifically, this holds if a shortest path between them (with respect to vertex weights p_i) has population greater than U.

For example, the vertex-weighted distance between vertices 1 and 9 in Figure 8 is $\text{dist}_{G,p}(1, 9) = 5$. So, if $U = 3$, then we can fix $x_{19} = 0$. This is a special case of the LCUT inequalities $x_{ab} \leq \sum_{c \in C} x_{cb}$ where the length-U a, b-separator is $C = \emptyset$.

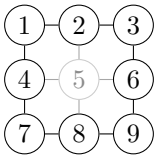

Figure 8: An example to illustrate U-fixing.

In the context of diagonal-fixing, we can do better. Instead of computing vertex-weighted distances in G, we can compute them in $G[V_j]$. That is, if $\text{dist}_{G[V_j],p}(i, j) > U$, then we can fix $x_{ij} = 0$. We call this U-fixing.

For example, recall the instance from Figure 8 in which the task is to split the 3x3 grid into three districts of equal size. Consider the ordering $(5, 2, 8, 1, 3, 4, 6, 7, 9)$. After diagonal-fixing, any vertex could be assigned to vertex 2 except for 5, i.e., $V_2 = V \setminus \{5\}$. However, the shortest vertex-weighted path from 8 to 2 in $G[V_2]$ has length 5, and thus vertex 8 cannot be assigned to vertex 2. This is despite the fact

that the vertex-weighted distance from 8 to 2 in G is suitable, $\text{dist}_{G,p}(2,8) = 3 \leq U$.

As reported in Table 5, an example of U-fixing for New Mexico is that Union County ($i = 33$), which is located in the state's northeast corner, cannot be assigned to Bernalillo County ($j = 19$), home to New Mexico's most populous city Albuquerque and located near the state's center. As Table 4 shows, U-fixing is helpful but not quite as helpful as diagonal-fixing and L-fixing. Nevertheless, we include U-fixing in our implementation given that its running time is negligible, requiring just n shortest path computations.

6.5 Z-fixing

Now, we fix some of the z variables. Recall that the primary constraint that variable z_e^j appears in is $x_{uj} - x_{vj} \leq z_e^j$ from model (7), and that we seek to minimize the sum of the z variables. Consequently, if we have already fixed x_{uj} to zero (or x_{vj} to one), then we can fix $z_e^j = 0$. We call this Z-fixing. As Table 4 reports, this fixes 83% or more of the z variables on the county-level instances, and 95% or more of the z variables on the tract-level instances.

7 Symmetry and Fixing for Labeling

In this section, we seek to improve the performance of the Labeling model. Since the Labeling model is already reasonably small, with just kn variables of the x_{ij} type, our primary aim is to deal with its symmetry. For this, we will apply an extended formulation for partitioning orbitopes that was proposed by Faenza and Kaibel [46]. (Later, we will experiment with the MIP solver's own symmetry handling techniques.) Our secondary aim is to reduce the size of the MIP by extending some of the variable fixing procedures that were developed in the previous section.

7.1 Symmetry handling via partitioning orbitope

Symmetry handling has been a hot topic in integer programming over the last 20 years [96], leading to ideas like isomorphism pruning [92, 93, 94, 95], orbital branching [108, 109], and orbitopal fixing [75], several of which are now used by state-of-the-art MIP solvers [111]. Another notable contribution in this area is the polyhedral study of packing and partitioning orbitopes performed by [76].

Of particular interest to us is the partitioning orbitope, which can be defined as the convex hull of 0-1 matrices with n rows and k columns that have precisely one "1" in each row and whose columns are sorted lexicographically. In this way, a partition of $V = [n]$ will have one canonical representation, eliminating the other $k! - 1$ alternative representations of the same plan. For example, consider the districting plan $\{\{5,2\},\{6\},\{4,1,3\}\}$. The district $\{5,2\}$ would be represented by the column vector $(0,1,0,0,1,0)^T$, the district $\{6\}$ by $(0,0,0,0,0,1)^T$, and the district $\{4,1,3\}$ by $(1,0,1,1,0,0)^T$. Sorting these column vectors lexicographically (decreasing) gives the canonical matrix representation for the Labeling variables:

$$x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In their work, Kaibel and Pfetsch [76] identify complete linear inequality descriptions of the packing and partitioning orbitopes. While the descriptions have exponentially many inequalities, the separation problem is linear-time solvable. Later, Faenza and Kaibel [46] give extended formulations for them that have size $O(kn)$, making the results easier to employ computationally. However, to our knowledge, no one has used them before. In our emails with symmetry-handling experts Faenza [46] and Pfetsch [73, 75, 76, 111], neither of them could recall anyone conducting experiments with these extended formulations [45, 110]. In our experiments, we will compare the extended formulation with Gurobi's symmetry handling techniques, which include orbitopal fixing [1].

Below, we give the partitioning orbitope extended formulation. We make some expository changes, e.g., we do not refer to network flows. We also add intuitive interpretations for the variables that were not provided in the original paper [46].

First, we identify a *root* for each district, given by its smallest indexed vertex (or, if an ordering is used, the district's earliest vertex in the ordering).

$$r_{ij} = \begin{cases} 1, \text{ if vertex } i \in V \text{ is the root of district } j \in [k], \\ 0, \text{ otherwise.} \end{cases}$$

For the example given above, the first district $\{4, 1, 3\}$ is rooted at vertex 1, the second district $\{5, 2\}$ is rooted at vertex 2, and the third district $\{6\}$ is rooted at vertex 6. With the root variables r_{ij}, this means that $r_{11} = r_{22} = r_{63} = 1$, while the others are zero. These same root variables can be used when imposing contiguity constraints. Other auxiliary variables keep track of certain recursions in the model.

$$s_{ij} = \begin{cases} 1, \text{ if vertex } i \in V \text{ is assigned to a district from set } \{j, j+1, \ldots, k\}, \\ 0, \text{ otherwise.} \end{cases}$$

$$w_{ij} = \begin{cases} 1, \text{ if root of district } j \in [k] \text{ belongs to set } \{1, 2, \ldots, i\}, \\ 0, \text{ otherwise.} \end{cases}$$

$$u_{ij} = \begin{cases} 1, \text{ if root of district } j \in [k] \text{ belongs to } \{1, 2, \ldots, i-1\} \text{ and} \\ \qquad \text{root of district } j+1 \text{ belongs to } \{i+1, i+2, \ldots, n\}, \\ 0, \text{ otherwise.} \end{cases}$$

The extended formulation for the partitioning orbitope uses the following constraints, where all out-of-range boundary values $s_{i,k+1}$ and $w_{0,j}$ and $r_{i,k+1}$ and $u_{n+1,j}$ equal zero, except for $r_{n+1,k+1} \equiv 1$.

$$x_{ij} = s_{ij} - s_{i,j+1} \qquad \forall i \in V,\ \forall j \in [k] \tag{14a}$$
$$r_{ij} = w_{ij} - w_{i-1,j} \qquad \forall i \in V,\ \forall j \in [k] \tag{14b}$$
$$r_{ij} \leq x_{ij} \qquad \forall i \in V,\ \forall j \in [k] \tag{14c}$$
$$s_{ij} \leq w_{ij} \qquad \forall i \in V,\ \forall j \in [k] \tag{14d}$$
$$r_{ij} + u_{ij} = r_{i+1,j+1} + u_{i+1,j} \qquad \forall i \in V,\ \forall j \in [k] \tag{14e}$$
$$r_{1,1} = 1 \tag{14f}$$
$$r_{ij},\ u_{ij},\ w_{ij},\ s_{ij} \in \{0,1\} \qquad \forall i \in V,\ \forall j \in [k]. \tag{14g}$$

Constraints (14a) relate the x and s variables, ensuring that vertex i is assigned to district j precisely when it is assigned to a district number in $\{j, j+1, \ldots, k\}$ but not in $\{j+1, \ldots, k\}$. Constraints (14b) relate the r and w variables, ensuring that vertex i roots district j precisely when district j's root belongs to $\{1, 2, \ldots, i\}$ but not to $\{1, 2, \ldots, i-1\}$. Constraints (14c) ensure that the root of a district belongs to said district. Constraints (14d) ensure that if vertex i is assigned to a district in $\{j, j+1, \ldots, k\}$, then district j is rooted at i or a vertex before i. Constraints (14e) are crucial "flow-balance" constraints in the original network interpretation. They impose that, if vertex i roots district j ($r_{ij} = 1$), then vertex $i+1$ either roots district $j+1$ or lies in the space "between" the roots of districts j and $j+1$ ($r_{i+1,j+1} + u_{i+1,j} = 1$). The same holds if vertex i lies between the roots of districts j and $j+1$ ($u_{ij} = 1$). Constraint (14f) initiates the flow that will be consumed via $r_{n+1,k+1} \equiv 1$. We remark that $\Theta(k^2)$ many of the variables will always equal zero (analogous to diagonal-fixing) and need not be created [46]. We also relax u, s, and w to be nonnegative continuous variables.

In Subsection 8.2, we will see that using model (14) sometimes reduces the number of branch-and-bound nodes by 100,000 and the solve time by 1,000 seconds.

7.2 L-fixing, U-fixing, and Z-fixing

Earlier, we proposed L-fixing for the Hess model. Here we propose essentially the same idea but for the Labeling model. One important condition is that we have root variables r_{ij} like those used in the partitioning orbitope extended formulation (14) and in the SCF model for imposing contiguity (5). As before, the choice of vertex ordering $(v_1, v_2, \ldots, v_n)$ is important, as it affects the resulting vertex subsets S_i as defined in (10). To maximize the number of fixings, we again prefer orderings that place a solution to problem (11) at the back. For this task, we use the same procedures outlined in Subsection 6.3. Now, if $p(S_i) < L$, then vertex i cannot root a feasible district, meaning that it is safe to fix $r_{ij} = 0$ for all $j \in [k]$.

More generally, suppose that a feasible solution B to problem (11) is placed at the back of the ordering, i.e., $B = \{v_q, v_{q+1}, \ldots, v_n\}$. Moreover, suppose that the districts are sorted lexicographically, as in the partitioning orbitope idea. Consider

a vertex $i = v_{q-1}$. If it were to root a district, then it must root district k, and we can fix $r_{ij} = 0$ for $j \leq k-1$. More generally, for vertex $i = v_{q-t}$ we can fix $r_{ij} = 0$ for all $j \leq k-t$. We call this L-fixing.

We also can extend U-fixing to the Labeling context. Recall that the first vertex in the ordering, $v = v_1$, must root the first district, i.e., $r_{v1} = 1$, by the partitioning orbitope idea. Other vertices u that are far from v, with $\text{dist}_{G,p}(v,u) > U$, cannot belong to this district, and we can safely fix $x_{u1} = 0$. Further, supposing that v_2 is far from v_1 (as is often true), then v_2 and v_1 cannot belong to the same district, and v_2 must be the root of district 2. Again, we can fix $x_{u2} = 0$ for vertices u that are far from v_2. These arguments continue, giving the following U-fixing procedure. In it, we use notation for the distance from vertex v to vertex subset S, $\text{dist}_{G,p}(v,S) = \min\{\text{dist}_{G,p}(v,u) \mid u \in S\}$, with convention that $\text{dist}_{G,p}(v,\emptyset) = \infty$.

- for $j = 1, 2, \ldots, k$ do
  - compute vertex-weighted distances from v_j, i.e., $\text{dist}_{G,p}(v_j, \cdot)$
  - if $\text{dist}_{G,p}(v_j, \{v_1, v_2, \ldots, v_{j-1}\}) \leq U$, then break
  - fix $r_{v_j,j} = 1$ and $x_{v_j,j} = 1$ and $x_{v_j,t} = 0$ for $t \in [k] \setminus \{j\}$
  - fix $r_{ij} = 0$ for other vertices $i \in V \setminus \{v_j\}$
  - for $u \in \{v_{j+1}, v_{j+2}, \ldots, v_n\}$ fix $x_{uj} = 0$ if $\text{dist}_{G,p}(v_j, u) > U$.

Finally, we can fix some z variables. That is, for a constraint $x_{uj} - x_{vj} \leq z_e^j$ of the Labeling model, if we have fixed $x_{uj} = 0$ or $x_{vj} = 1$, then we can fix z_e^j to zero. Also, if we have fixed $x_{uj} = 1$ and $x_{vj} = 0$, then we can fix z_e^j to one.

Table 6 reports on the performance of the fixing procedures for the Labeling model. The results show that most r variables are fixed, with 88% or more fixed on county-level instances, and 96% or more fixed on tract-level instances. Meanwhile, few x variables are fixed: at most 34% on the county-level instances and 0% on the tract-level instances. Similarly, few z variables are fixed: at most 29% on county-level instances, and 0% or 1% on tract-level instances. This is expected given the small number of x variables that were fixed.

8 Final Computational Experiments

In this section, we conduct final computational experiments. Our aim is to shed light on the following questions.

1. Does the extended objective from Section 4 help? How much does it strengthen the root LP bound in practice? What is its overall effect on MIP solve time? Do the benefits of a stronger model outweigh the cost of its larger size?
2. Which contiguity constraints perform best (among LCUT, SCF, and SHIR)? Does the answer depend on the base model (Hess vs. Labeling)? Does the answer depend on the symmetry handling technique (solver default vs. aggressive symmetry handling vs. partitioning orbitope extended formulation)?

Table 6: Number and percentage of fixings for Labeling model with contiguity.

			r_{ij}		x_{ij}		z^j_{uv}	
state	n	k	#	%	#	%	#	%
ME	16	2	28	88	2	6	5	7
NM	33	3	93	94	34	34	69	29
ID	44	2	84	95	2	2	5	2
WV	55	3	151	92	4	2	14	4
LA	64	6	353	92	38	10	80	8
AL	67	7	426	91	111	24	287	24
AR	75	4	269	90	7	2	20	3
OK	77	5	359	93	97	25	250	26
MS	82	4	291	89	7	2	20	2
NE	93	3	271	97	11	4	13	2
IA	99	4	356	90	7	2	17	2
KS	105	4	409	97	17	4	34	3
NH	295	2	577	98	2	0	7	0
ID	298	2	588	99	2	0	11	1
ME	358	2	706	99	2	0	8	0
WV	484	3	1,418	98	4	0	18	0
NM	499	3	1,469	98	4	0	24	1
NE	532	3	1,558	98	4	0	13	0
UT	588	4	2,258	96	7	0	37	1
MS	664	4	2,568	97	7	0	23	0
AR	686	4	2,647	96	7	0	23	0
NV	687	4	2,648	96	7	0	42	1

3. Overall, what is the fastest MIP approach for minimizing cut edges? What size instances can it solve? How much faster is it than a naïve approach (without our proposed arsenal of MIP tricks)?

These questions are answered in the following three subsections, respectively. Recall that our PC, MIP solver, and test instances were discussed in Section 3.

8.1 Evaluating extended cut edges objective

Here, we analyze the impacts of the extended objective from Section 4. We do not impose contiguity constraints in this section; their impact will be evaluated in the next subsection.

First, we consider the effect of the extended objective in the context of the Hess base model. We employ diagonal-fixing and L-fixing (without contiguity), as discussed previously, and use Gurobi's default symmetry handling. Note that U-fixing does not really apply when contiguity is not imposed. Results with and without constraints (7) are provided in Table 7. As the table shows, the extended

objective often improves the LP bound substantially. For example, the LP bound strengthens from 5.73 to 22.30 for Louisiana, from 13.38 to 29.05 for Alabama, and from 3.58 to 14.66 for Iowa. With these stronger bounds, the MIPs are solved more quickly. For example, the MIP solve time improves from 2,093 seconds to 307 seconds for Iowa. Further, neither Louisiana nor Alabama were solved within the one-hour time-limit using the original objective, but with aid of the extended objective they were solved in 1,003 and 1,109 seconds, respectively.

Table 7: Impact of extended objective on the Hess base model (w/o contiguity constraints) at county and tract levels under a 3,600-second time-limit (TL). We report the LP relaxation bound (LP), the number of branch-and-bound nodes visited (B&B), and the time to solve the MIP in seconds (time).

			w/o extended objective			w/ extended objective		
state	n	k	LP	B&B	time	LP	B&B	time
ME	16	2	2.74	1	0.05	3.62	1	0.08
NM	33	3	6.43	1	0.52	10.97	1	0.23
ID	44	2	2.53	1	0.85	4.27	29	1.24
WV	55	3	4.22	1,282	13.60	9.41	1,001	5.93
LA	64	6	5.73	136,218	TL	22.30	257,754	1,002.87
AL	67	7	13.38	141,054	TL	29.05	289,115	1,108.70
AR	75	4	6.40	58,002	1,003.25	15.92	25,446	146.39
OK	77	5	12.62	40,662	510.67	21.36	2,194	32.34
MS	82	4	4.16	90,920	1,218.58	12.88	36,837	264.70
NE	93	3	3.19	661	103.64	9.71	960	21.53
IA	99	4	3.58	81,034	2,092.95	14.66	21,280	307.11
KS	105	4	4.76	34,992	521.03	13.45	18,153	236.76
NH	295	2	-	1	TL	7.44	1,626	2,426.24
ID	298	2	-	1	TL	5.31	114	1,655.01
ME	358	2	-	0	TL	-	3	TL
WV	484	3	-	0	TL	-	0	TL
NM	499	3	-	0	TL	-	0	TL
NE	532	3	-	0	TL	-	1	TL
UT	588	4	-	0	TL	-	0	TL
MS	664	4	-	0	TL	22.27	0	TL
AR	686	4	-	0	TL	-	0	TL
NV	687	4	-	0	TL	-	0	TL

Interestingly, while two tract-level MIPs could be solved with the extended objective, *none* of the tract-level *LP relaxations* could be solved with the original objective. This is perhaps surprising given that the extended objective uses more variables! Inspecting the log files, we see why. For example, when solving the tract-level LP for New Hampshire, the solver spends 112 seconds in barrier and another 92 seconds for the primal and dual push phases; however, the final simplex cleanup

continued until the one-hour time-limit was reached. Meanwhile, when the extended objective is used, barrier takes 6 seconds, the primal and dual push phases take 3 seconds, and the final simplex cleanup takes 236 seconds. For whatever reason, the LP solver has an easier time with the extended objective.

Next, we consider the effect of the extended objective on the Labeling base model. Results with and without the constraints of Ferreira et al. [47] are provided in Table 8. As before, the extended objective often improves the LP bound significantly. For example, the LP bound strengthens from 5.49 to 16.29 for Louisiana, from 11.56 to 19.35 for Alabama, and from 5.36 to 10.03 for Iowa, allowing the MIPs to be solved more quickly. For example, the MIP solve time improves from 262 seconds to 24 seconds for Iowa. Again, neither Louisiana nor Alabama were solved within the one-hour time-limit using the original objective, but with aid of the extended objective they were solved in 1,136 and 2,871 seconds, respectively, and visited substantially fewer branch-and-bound nodes. Another notable instance is Nebraska at the tract-level, whose MIP solve time improves from 3,585 seconds to 59 seconds when the extended objective is used. The number of branch-and-bound nodes also drops from 477,860 to 3,877.

Comparing the results from Tables 7 and 8, we see that the Labeling model typically solves more quickly than the Hess model, whether or not the extended objective is used. The differences are most pronounced on the large, tract-level instances when k is very small (e.g., $k = 2$). In these cases, the Labeling base model uses approximately n variables, while the Hess base model uses approximately n^2 variables. For example, when applying the extended objective to New Hampshire ($k = 2$) at the tract-level, the Labeling base model has 1,852 variables remaining after presolve, while the Hess base model has 107,933 variables remaining after presolve. We think this explains the drastic differences in MIP solve time. In contrast, when k is larger, the story is a little different. For example, when applying the extended objective to Alabama ($k = 7$) at the county-level, the Labeling base model has 1,343 variables after presolve, while the Hess base model is only slightly larger with 5,520 variables after presolve, putting them in similar territory. In fact, the MIP solves *more quickly* with the Hess base model than with the Labeling base model (1,109 seconds versus 2,871 seconds). In the next subsection, as contiguity is imposed, many more Hess variables will be fixed via L-fixing. This diminishes the huge size advantage that the Labeling model has enjoyed.

Finally, we observe that the cut edges objective does not achieve contiguity "for free," despite some researchers' beliefs to the contrary. Figure 9 shows *tract-level* districting plans for Idaho and West Virginia that, despite having a minimum number of cut edges, are not contiguous.

8.2 Evaluating contiguity constraints and symmetry handling

Here, we analyze the impact of the different approaches for imposing contiguity and symmetry handling. All implementations use the fixing procedures L-fixing, U-fixing, diagonal-fixing, and Z-fixing, as well as the extended objective (due to its strong performance in the previous subsection).

Table 8: Impact of extended objective on the Labeling base model (w/o contiguity constraints) at county and tract levels under a 3,600-second time-limit (TL). We report the LP relaxation bound (LP), the number of branch-and-bound nodes visited (B&B), and the time to solve the MIP in seconds (time).

			w/o extended objective			w/ extended objective		
state	n	k	LP	B&B	time	LP	B&B	time
ME	16	2	2.71	1	0.02	2.71	1	0.02
NM	33	3	5.75	92	0.14	6.63	17	0.14
ID	44	2	3.32	1	0.03	3.32	1	0.03
WV	55	3	5.55	6,101	1.77	7.09	599	0.36
LA	64	6	5.49	2,158,617	TL	16.29	433,484	1,136.13
AL	67	7	11.56	1,515,101	TL	19.35	707,025	2,871.13
AR	75	4	5.69	426,704	353.21	9.04	39,142	29.09
OK	77	5	13.95	83,723	108.74	14.94	3,778	7.88
MS	82	4	6.46	488,879	365.17	7.72	36,268	27.12
NE	93	3	3.87	706	1.62	6.13	584	0.89
IA	99	4	5.36	228,583	262.03	10.03	16,294	23.56
KS	105	4	7.91	103,634	136.01	10.23	6,206	11.38
NH	295	2	3.50	3,350	3.24	3.50	1,036	1.50
ID	298	2	2.27	638	1.47	2.27	302	0.81
ME	358	2	4.00	123	0.80	4.00	76	0.78
WV	484	3	5.74	72,476	579.40	7.05	8,468	92.79
NM	499	3	7.02	23,006	213.15	12.05	4,287	45.69
NE	532	3	3.67	477,860	3,584.86	5.34	3,877	58.68
UT	588	4	10.37	32,365	TL	15.37	76,338	TL
MS	664	4	7.85	32,379	TL	13.01	47,501	TL
AR	686	4	4.60	17,974	TL	8.77	28,520	TL
NV	687	4	9.94	19,456	TL	17.55	36,848	TL

When implementing the LCUT models, we add violated inequalities in a callback procedure and thus need to invoke Gurobi's `LazyConstraints` parameter. As in our previous work [131], we separate only *integer* points x^* that represent disconnected districting plans. For this, we use the linear-time algorithm of Fischetti et al. [50] to identify a minimal a, b-separator and then chisel it down to a minimal length-U a, b-separator with a simple procedure [131]. See our code for the full details.

First, we consider the effect of the different contiguity constraints in the context of the Hess base model. Results are provided in Table 9. All contiguity models solve the county-level instances, with the most difficult instance being Alabama, which requires roughly 20 minutes to solve. The county-level times are roughly comparable across the different contiguity models, with each taking its turn as the noticeably fastest method: the SCF model for KS, the SHIR model for LA; and the LCUT model for AL, AR, MS, and IA. If a "winner" had to be chosen for

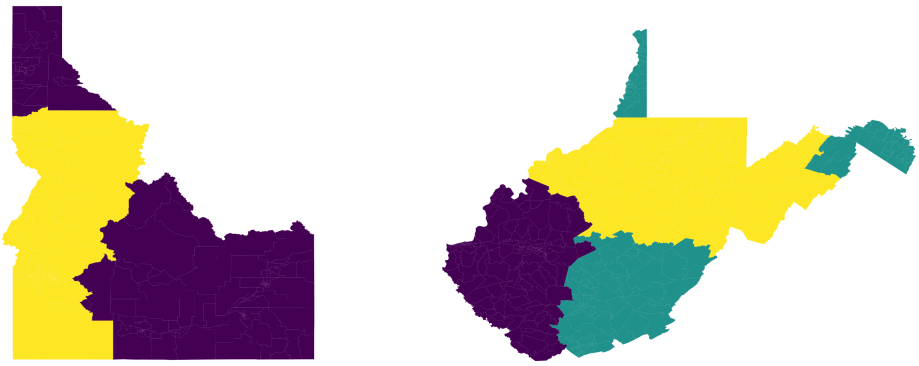

Figure 9: Contiguity does not come "for free" for ID and WV at the tract-level.

the county-level instances, we might give the award to LCUT. On the tract-level instances, SCF takes a lead over LCUT (e.g., on WV and NM), and LCUT takes a lead over SHIR (e.g., on WV, NM, and NE). Overall, we think the three different contiguity models are all reasonable choices for the Hess model on these instances.

Next, we consider the effect of the different contiguity constraints in the context of the Labeling base model. We recognize that the LCUT model, which requires the use of callbacks and the `LazyConstraints` parameter, will deactivate many or all of Gurobi's built-in symmetry handling techniques. For this reason, we were interested to see which symmetry handling technique would perform best for it. Table 10 provides results for the Label-LCUT model under Gurobi's default symmetry handling (default), Gurobi's aggressive symmetry handling (aggressive), and the extended formulation for partitioning orbitopes (orbitope). As expected, there is little difference between the default and aggressive settings—all of which could be attributed to noise. Introducing the orbitope setting into the mix, we see that it is sometimes faster than the default setting: 388 seconds versus 504 seconds on LA, and 24 seconds versus 228 seconds on NM (tract). Sometimes it is slower than default: 27 seconds versus 10 seconds on KS, and 163 seconds versus 131 seconds on NE (tract). Overall, we see a slight advantage to the orbitope setting.

We now consider the SCF model. Since the entire SCF model is provided to the MIP solver a priori, we had hope that Gurobi's built-in symmetry handling techniques would be effective. However, we found that the orbitope setting was actually *more* effective than the default setting and the aggressive setting, as Table 11 shows. This could be explained by our variable fixing, which disturbs the symmetry, leaving nothing for the MIP solver to exploit. Consequently, we reran the results, disabling any variable fixing procedure that disturbed the symmetry between the district labels: diagonal-fixing (lines 109–114 of `fixing.py`), a portion of L-fixing (lines 162–167), and U-fixing (lines 219–254). With these changes, the results were noticeably worse, with LA and AL failing to solve within the one-hour time-limit. Others, like Hojny et al. [72], make similar observations when using the SCIP solver; built-in symmetry handling is fragile and is disrupted by other tricks.

Table 9: Results for Hess base model with different contiguity models. We report the lower bound (LB) and upper bound (UB) at termination under a 3,600-second time-limit (TL).

			Hess-LCUT			Hess-SCF			Hess-SHIR		
state	n	k	LB	UB	time	LB	UB	time	LB	UB	time
ME	16	2	16	16	0.19	16	16	0.11	16	16	0.11
NM	33	3	17	17	0.28	17	17	0.12	17	17	0.12
ID	44	2	10	10	0.31	10	10	0.16	10	10	0.25
WV	55	3	23	23	3.04	23	23	1.69	23	23	2.14
LA	64	6	49	49	209.87	49	49	433.77	49	49	91.50
AL	67	7	55	55	1,125.44	55	55	1,200.22	55	55	1,252.13
AR	75	4	33	33	45.86	33	33	54.05	33	33	65.19
OK	77	5	40	40	9.43	40	40	10.06	40	40	12.89
MS	82	4	34	34	107.80	34	34	128.20	34	34	160.19
NE	93	3	19	19	2.47	19	19	4.86	19	19	4.87
IA	99	4	33	33	72.61	33	33	96.34	33	33	192.49
KS	105	4	32	32	62.00	32	32	20.61	32	32	29.30
NH	295	2	26	26	110.69	26	26	102.73	26	26	143.98
ID	298	2	17	17	22.43	17	17	33.25	17	17	23.84
ME	358	2	20	20	80.54	20	20	70.52	20	20	49.70
WV	484	3	43	43	1,142.36	43	43	656.71	43	43	2,239.97
NM	499	3	43	43	695.74	43	43	385.64	28	43	TL
NE	532	3	30	44	TL	40	44	TL	17	48	TL
UT	588	4	30	90	TL	31	90	TL	26	90	TL
MS	664	4	25	65	TL	27	64	TL	24	68	TL
AR	686	4	19	76	TL	19	78	TL	19	78	TL
NV	687	4	27	89	TL	29	86	TL	26	89	TL

Last, we turn to the SHIR model. Again, the SHIR model is provided in its entirety to the MIP solver a priori, giving us hope that the MIP solver's built-in symmetry handling techniques could shine. The results in Table 12 are mixed. First, we observe that the default and aggressive settings perform similarly, typically visiting the same number of branch-and-bound nodes and solving in about the same time. It appears that the aggressive setting did not change the solver's behavior. Introducing the orbitope setting into the mix, we see that it sometimes performs much better than the default setting, e.g., 956 seconds versus 1648 seconds for AL. Other times it performs worse, e.g., 321 seconds versus 116 seconds for LA, and 335 seconds versus 157 seconds for WV (tract). Overall, there might be a slight advantage to the orbitope setting on these instances.

In Table 13, we summarize the performance of the Labeling model using the different contiguity models. In each case, symmetry is handled with the orbitope setting. The contiguity models solve the exact same set of instances within the one-hour time-limit, with AL again being the most difficult county-level instance and LA the next hardest. Each contiguity model takes its turn as the fastest method:

Table 10: Comparison of symmetry handling methods for Label-LCUT.

			default symmetry		aggressive symmetry		orbitope	
state	n	k	#B&B	MIP time	#B&B	MIP time	#B&B	MIP time
ME	16	2	427	0.12	427	0.17	1080	0.16
NM	33	3	11	0.11	11	0.11	7	0.09
ID	44	2	41	0.11	41	0.11	16	0.12
WV	55	3	3,940	1.78	3,940	1.80	1,785	1.05
LA	64	6	423,363	504.37	423,363	499.04	272,631	387.96
AL	67	7	995,220	1,005.77	995,220	999.32	802,391	1,017.77
AR	75	4	38,142	28.43	38,142	28.49	26,771	23.40
OK	77	5	10,211	8.82	10,211	8.82	9,037	8.50
MS	82	4	77,287	38.83	77,287	38.45	86,016	54.18
NE	93	3	391	0.67	391	0.68	782	1.47
IA	99	4	21,204	27.68	21,204	27.98	19,379	29.18
KS	105	4	5,311	10.15	5,311	10.03	11,565	27.28
NH	295	2	3,760	3.42	3,760	3.44	2,461	3.50
ID	298	2	2,531	2.03	2,531	2.00	563	1.73
ME	358	2	6,061	4.27	6,061	4.33	2,123	2.53
WV	484	3	12,978	140.67	12,978	145.72	12,342	159.04
NM	499	3	14,585	228.20	14,585	230.63	1,703	24.20
NE	532	3	8,337	131.34	8,337	130.46	8,451	163.29
UT	588	4	83,635	TL	83,873	TL	80,652	TL
MS	664	4	57,365	TL	57,588	TL	38,540	TL
AR	686	4	33,883	TL	34,144	TL	19,376	TL
NV	687	4	46,361	TL	44,986	TL	32,611	TL

LCUT for AR, IA, NM (tract), and NE (tract); SCF for OK, KS, and WV (tract); and SHIR for LA. Overall, it appears that LCUT takes a slight lead over the other contiguity models, although they all seem to be reasonable choices.

8.3 Identifying fastest MIP approach and limitations

Here we try to identify the fastest MIP approach and its limitations. To this end, Table 14 summarizes the solve times that were reported in Tables 9 and 13. On the county-level instances, the results are mixed. For example, Hess-SHIR is fastest on LA, Labeling-SHIR on AL, Labeling-LCUT on AR and IA, Labeling-SCF on MS and KS. Generally, the Labeling implementations are faster, dominating the Hess implementations on the county-level instances MS and IA. This dominance becomes more pronounced on the tract-level instances, with the Labeling implementations often being 10x-50x faster than their Hess counterparts. Notably, each of the Labeling implementations solve NE (tract) in under 7 minutes, but none of the Hess implementations solve it within one hour.

Table 11: Comparison of symmetry handling methods for Label-SCF.

			default symmetry		aggressive symmetry		orbitope	
state	n	k	#B&B	MIP time	#B&B	MIP time	#B&B	MIP time
ME	16	2	218	0.09	218	0.11	195	0.11
NM	33	3	1	0.06	1	0.09	1	0.05
ID	44	2	14	0.12	14	0.14	1	0.05
WV	55	3	2,851	1.20	2,851	1.21	1,476	0.83
LA	64	6	636,673	1,264.97	426,993	972.39	265,550	494.32
AL	67	7	808,085	2,782.71	749,042	2,530.60	516,476	1,539.34
AR	75	4	36,091	35.47	36,091	36.55	18,842	30.00
OK	77	5	2,420	3.95	2,420	3.93	2,877	3.92
MS	82	4	90,911	146.70	86,403	146.37	33,435	32.10
NE	93	3	99	0.47	99	0.52	140	0.59
IA	99	4	33,537	54.09	33,537	54.82	23,266	35.64
KS	105	4	6,892	13.25	6,892	13.17	5,989	12.77
NH	295	2	5,574	7.66	5,574	7.84	1,222	2.84
ID	298	2	505	1.56	505	1.55	182	1.00
ME	358	2	102	1.13	102	1.19	131	1.28
WV	484	3	25,274	322.91	25,274	334.10	7,681	153.19
NM	499	3	6,991	119.40	6,991	115.38	5,742	97.74
NE	532	3	16,503	286.07	16,503	272.39	9,179	208.36
UT	588	4	48,727	TL	46,941	TL	43,353	TL
MS	664	4	37,759	TL	33,082	TL	30,272	TL
AR	686	4	20,210	TL	19,848	TL	27,440	TL
NV	687	4	33,816	TL	34,276	TL	33,025	TL

Comparing the times from Table 14 against those of the naïve implementation from Table 1, we see huge improvements. For example, the naïve Hess implementation could not solve WV (county) within one hour (nor any of the larger instances), but now it takes 2 or 3 seconds. Contiguity is also now enforced. The Labeling model, which was previously unable to solve LA and AL, can now solve them within 20 minutes (and with contiguity). Key to these improvements are all the tricks from the MIP arsenal that we use. Based on the experiments from the previous subsection, it appears that symmetry handling and contiguity constraints have less impact on solve time than the heuristic warm start, extended objective, and variable fixing.

9 Conclusion and Future Work

In this paper, we study a stylized redistricting problem in which the task is to split up a graph into a prescribed number of contiguous districts, each satisfying population balance constraints, with the objective of minimizing the number of "cut" edges between districts. First, we observe that a straightforward MIP model,

Table 12: Comparison of symmetry handling methods for Label-SHIR.

			default symmetry		aggressive symmetry		orbitope	
state	n	k	#B&B	MIP time	#B&B	MIP time	#B&B	MIP time
ME	16	2	153	0.08	153	0.09	131	0.11
NM	33	3	1	0.06	1	0.06	1	0.05
ID	44	2	16	0.12	16	0.12	1	0.08
WV	55	3	1,072	0.97	1,072	0.89	1,287	1.19
LA	64	6	70,363	115.63	70,363	114.68	113,081	321.16
AL	67	7	312,635	1,647.99	312,635	1,640.54	225,733	956.30
AR	75	4	26,404	44.08	26,404	43.84	16,177	39.90
OK	77	5	1,903	4.14	1,903	4.12	1,072	3.59
MS	82	4	53,258	71.88	53,258	73.23	47,384	71.10
NE	93	3	104	0.75	104	0.87	62	0.97
IA	99	4	31,080	76.47	31,080	76.44	29,756	78.96
KS	105	4	8,887	28.34	8,887	28.24	8,955	26.95
NH	295	2	1,836	9.77	1,836	9.64	2,184	12.58
ID	298	2	166	1.98	166	1.97	146	1.77
ME	358	2	156	3.13	156	3.02	35	1.69
WV	484	3	4,768	157.27	4,768	157.36	13,099	335.30
NM	499	3	1,738	68.44	1,738	68.50	1,160	62.83
NE	532	3	13,507	443.71	13,507	443.44	7,333	393.22
UT	588	4	19,933	TL	19,929	TL	16,096	TL
MS	664	4	9,370	TL	9,386	TL	8,000	TL
AR	686	4	4,837	TL	4,799	TL	3,432	TL
NV	687	4	7,044	TL	6,996	TL	8,435	TL

the Labeling model, which has been proposed in the previous literature, is unable to solve all county-level instances in the USA. Another model, the Hess model, is also ill-suited for this task when a naïve implementation is used. This is partially a consequence of their weak LP relaxations and model symmetry. Moreover, they often generate disconnected districting plans—even at the census tract level—if contiguity is not explicitly enforced.

In response, we use various MIP tricks to speed up the computations: a strong extended formulation for the cut edges objective [47], heuristic warm starts via GerryChain [99], symmetry handling via partitioning orbitopes [46] and diagonal-fixing [20], and other powerful variable fixing techniques that allow us to fix 95% or more of the Hess variables on tract-level instances. To our knowledge, we are the first to implement and use Faenza and Kaibel's extended formulation for partitioning orbitopes [46]. The newly proposed L-fixing procedure is also crucial to making the Hess base model a reasonable choice on county-level instances. Nevertheless, we ultimately find that the Labeling base model performs better than the Hess base model on large, tract-level instances when the number of districts is small. It

Table 13: Results for Labeling base model with different contiguity models. We report the lower bound (LB) and upper bound (UB) at termination under a 3,600-second time-limit (TL). All experiments here use the orbitope setting.

			Label-LCUT			Label-SCF			Label-SHIR		
state	n	k	LB	UB	time	LB	UB	time	LB	UB	time
ME	16	2	16	16	0.17	16	16	0.12	16	16	0.09
NM	33	3	17	17	0.09	17	17	0.06	17	17	0.06
ID	44	2	10	10	0.12	10	10	0.05	10	10	0.08
WV	55	3	23	23	1.11	23	23	1.03	23	23	1.23
LA	64	6	49	49	394.16	49	49	512.65	49	49	320.69
AL	67	7	55	55	1,019.40	55	55	1,269.54	55	55	956.12
AR	75	4	33	33	23.54	33	33	31.79	33	33	40.38
OK	77	5	40	40	8.50	40	40	4.11	40	40	6.78
MS	82	4	34	34	54.80	34	34	32.06	34	34	70.84
NE	93	3	19	19	1.44	19	19	0.62	19	19	0.91
IA	99	4	33	33	29.79	33	33	35.59	33	33	79.24
KS	105	4	32	32	27.21	32	32	13.03	32	32	27.09
NH	295	2	26	26	3.53	26	26	2.83	26	26	12.58
ID	298	2	17	17	1.77	17	17	1.02	17	17	1.70
ME	358	2	20	20	2.72	20	20	1.29	20	20	1.66
WV	484	3	43	43	158.64	43	43	151.84	43	43	334.49
NM	499	3	43	43	24.38	43	43	96.35	43	43	62.80
NE	532	3	44	44	163.78	44	44	207.06	44	44	392.56
UT	588	4	66	79	TL	63	79	TL	61	79	TL
MS	664	4	45	63	TL	47	63	TL	41	63	TL
AR	686	4	39	70	TL	40	72	TL	30	70	TL
NV	687	4	55	77	TL	55	77	TL	51	77	TL

allows us to solve instances with up to 532 census tracts to optimality in a one-hour time-limit. When it comes to imposing contiguity constraints, we find that the single-commodity flow formulation (SCF) of Hojny et al. [72], the multi-commodity flow formulation (SHIR) of Shirabe [104, 124, 125], and the LCUT model of Validi et al. [131] are all reasonable choices, with the SCF model and LCUT model sometimes taking a slight lead, depending on the circumstances. Our data, code, and results are all publicly available in a GitHub repository: `https://github.com/hamidrezavalidi/Political-Districting-to-Minimize-Cut-Edges`.

We mention two opportunities for future work. Our procedures are able to handle all county-level instances, but only some tract-level instances ($n \leq 532$). Meanwhile, we are aware of some other, related districting instances (`nj1`, `nj2`, `nj3`) that were generated by Jonathan Eckstein for MIPLIB 2017 that remain unsolved [43, 55]. It would be nice to develop improved procedures for handling these large instances. A big obstacle to overcome are the weak LP bounds that cut edge

Table 14: Solve times under a 3600-second limit for the Hess and Labeling base models with different contiguity models (reproduced from Tables 9 and 13).

			Hess base model			Labeling base model		
state	n	k	LCUT	SCF	SHIR	LCUT	SCF	SHIR
ME	16	2	0.19	0.11	0.11	0.17	0.12	0.09
NM	33	3	0.28	0.12	0.12	0.09	0.06	0.06
ID	44	2	0.31	0.16	0.25	0.12	0.05	0.08
WV	55	3	3.04	1.69	2.14	1.11	1.03	1.23
LA	64	6	209.87	433.77	91.50	394.16	512.65	320.69
AL	67	7	1,125.44	1,200.22	1,252.13	1,019.40	1,269.54	956.12
AR	75	4	45.86	54.05	65.19	23.54	31.79	40.38
OK	77	5	9.43	10.06	12.89	8.50	4.11	6.78
MS	82	4	107.80	128.20	160.19	54.80	32.06	70.84
NE	93	3	2.47	4.86	4.87	1.44	0.62	0.91
IA	99	4	72.61	96.34	192.49	29.79	35.59	79.24
KS	105	4	62.00	20.61	29.30	27.21	13.03	27.09
NH	295	2	110.69	102.73	143.98	3.53	2.83	12.58
ID	298	2	22.43	33.25	23.84	1.77	1.02	1.70
ME	358	2	80.54	70.52	49.70	2.72	1.29	1.66
WV	484	3	1,142.36	656.71	2,239.97	158.64	151.84	334.49
NM	499	3	695.74	385.64	TL	24.38	96.35	62.80
NE	532	3	TL	TL	TL	163.78	207.06	392.56
UT	588	4	TL	TL	TL	TL	TL	TL
MS	664	4	TL	TL	TL	TL	TL	TL
AR	686	4	TL	TL	TL	TL	TL	TL
NV	687	4	TL	TL	TL	TL	TL	TL

models provide. For example, one could use semidefinite relaxations [44] or develop a branch-and-price algorithm for a set partitioning model that uses a binary variable for each possible district [54, 97, 98]. As noted to us by Eckstein [43], doing these tasks well are nontrivial, and we consider it to be outside our scope. Another opportunity is to extend our work to minimize the Polsby-Popper score [113], which is a nonlinear expression relating to district area and perimeter—the latter of which can be viewed as *weighted* cut edges.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1942065. Hamid thanks Moon Duchin for bringing cut edges to his attention as a redistricting compactness measure during VRDI 2019.

References

[1] Tobias Achterberg. Symmetry breaking algorithm in Gurobi. `https://support.gurobi.com/hc/en-us/community/posts/360050295511-Symmetry-Breaking-Algorithm-in-Gurobi`, 2020. Accessed: 2021-02-22.

[2] William T Adler and Samuel S-H Wang. Response to Cho and Liu, "Sampling from complicated and unknown distributions: Monte Carlo and Markov chain Monte Carlo methods for redistricting". *Physica A: Statistical Mechanics and its Applications*, 516:591–593, 2019.

[3] Zacharie Alès and Arnaud Knippel. The k-partitioning problem: Formulations and branch-and-cut. *Networks*, 76(3):323–349, 2020.

[4] Micah Altman. The computational complexity of automated redistricting: Is automation the answer? *Rutgers Computer & Tech. LJ*, 23:81, 1997.

[5] Micah Altman and Michael McDonald. Redistricting by formula: An Ohio reform experiment. *American Politics Research*, 46(1):103–131, 2018.

[6] Micah Altman, Michael P McDonald, et al. BARD: Better automated redistricting. *Journal of Statistical Software*, 42(4):1–28, 2011.

[7] Kenneth I Appel and Wolfgang Haken. *Every planar map is four colorable*, volume 98. American Mathematical Society, 1989.

[8] Fernando Bacao, Victor Lobo, and Marco Painho. Applying genetic algorithms to zone design. *Soft Computing*, 9(5):341–348, 2005.

[9] David A Bader, Henning Meyerhenke, Peter Sanders, and Dorothea Wagner. *Graph partitioning and graph clustering*, volume 588. American Mathematical Society Providence, RI, 2013.

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

[11] Richard Barnes and Justin Solomon. Gerrymandering and compactness: Implementation flexibility and abuse. *Political Analysis*, 2020. To appear.

[12] Michael Bastubbe and Marco E Lübbecke. A branch-and-price algorithm for capacitated hypergraph vertex separation. *Mathematical Programming Computation*, 12(1):39–68, 2020.

[13] Amariah Becker and Justin Solomon. Redistricting algorithms, 2020.

[14] Walid Ben-Ameur, Mohamed-Ahmed Mohamed-Sidi, and José Neto. The k-separator problem: polyhedra, complexity and approximation results. *Journal of Combinatorial Optimization*, 29(1):276–307, 2015.

[15] Charles-Edmond Bichot and Patrick Siarry, editors. *Graph Partitioning*. John Wiley & Sons, 2013.

[16] Ralf Borndörfer, Carlos E Ferreira, and Alexander Martin. Decomposing matrices into blocks. *SIAM Journal on Optimization*, 9(1):236–269, 1998.

[17] Burcin Bozkaya, Erhan Erkut, and Gilbert Laporte. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*, 144(1):12–26, 2003.

[18] Michelle H Browdy. Simulated annealing: an improved computer model for political redistricting. *Yale Law & Policy Review*, 8(1):163–179, 1990.

[19] Charles S Bullock III. *Redistricting: The most political activity in America*. Rowman & Littlefield Publishers, 2010.

[20] Manoel Campêlo, Victor A Campos, and Ricardo C Corrêa. On the asymmetric representatives formulation for the vertex coloring problem. *Discrete Applied Mathematics*, 156(7):1097–1111, 2008.

[21] Rodolfo Carvajal, Miguel Constantino, Marcos Goycoolea, Juan Pablo Vielma, and Andrés Weintraub. Imposing connectivity constraints in forest planning models. *Operations Research*, 61(4):824–836, 2013.

[22] Wendy K Tam Cho and Yan Y Liu. Sampling from complicated and unknown distributions: Monte Carlo and Markov chain Monte Carlo methods for redistricting. *Physica A: Statistical Mechanics and its Applications*, 506:170–178, 2018.

[23] Sunil Chopra and Mendu R Rao. The partition problem. *Mathematical Programming*, 59(1-3):87–115, 1993.

[24] Sunil Chopra and MR Rao. Facets of the k-partition polytope. *Discrete Applied Mathematics*, 61(1):27–48, 1995.

[25] Christine Chou, Steven O Kimbrough, Frederic H Murphy, John Sullivan-Fedock, and C Jason Woodard. On empirical validation of compactness measures for electoral redistricting and its significance for application of models in the social sciences. *Social Science Computer Review*, 32(4):534–543, 2014.

[26] Vincent Cohen-Addad, Philip N Klein, and Neal E Young. Balanced centroidal power diagrams for redistricting. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 389–396. ACM, 2018.

[27] Vincent Cohen-Addad, Philip N. Klein, and Daniel Marx. On the computational tractability of a geographic clustering problem arising in redistricting. *arXiv preprint arXiv:2009.00188*, 2020.

[28] Michele Conforti, MR Rao, and Antonio Sassano. The equipartition polytope. II: Valid inequalities and facets. *Mathematical Programming*, 49(1):71–90, 1990.

[29] Denis Cornaz, Fabio Furini, Mathieu Lacroix, Enrico Malaguti, A Ridha Mahjoub, and Sébastien Martin. The vertex k-cut problem. *Discrete Optimization*, 31:8–28, 2019.

[30] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[31] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[32] Mark S Daskin and Emily L Tucker. The trade-off between the median and range of assigned demand in facility location models. *International Journal of Production Research*, 56(1-2):97–119, 2018.

[33] Daryl DeFord. Dual graphs for 2010 census units, 2021. `https://people.csail.mit.edu/ddeford/dual_graphs.html`.

[34] Daryl DeFord and Moon Duchin. Redistricting reform in virginia: Districting criteria in context. *Virginia Policy Review*, 12(2):120–146, 2019.

[35] Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: A family of markov chains for redistricting. *Harvard Data Science Review*, 3 2021. https://hdsr.mitpress.mit.edu/pub/1ds8ptxu.

[36] Daryl DeFord, Hugo Lavenant, Zachary Schutzman, and Justin Solomon. Total variation isoperimetric profiles. *SIAM Journal on Applied Algebra and Geometry*, 3(4):585–613, 2019.

[37] Michel Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17(4):981–1000, 1992.

[38] Rodney G Downey and Michael R Fellows. *Fundamentals of Parameterized Complexity*, volume 4. Springer, 2013.

[39] Matthew Dube and Jesse Clark. Beyond the circle: Measuring district compactness using graph theory. In *Northeast Political Science Association Conference*, 2016.

[40] Moon Duchin and Steven Strogatz. Moon Duchin on fair voting and random walks. Quanta Magazine, 2020. `https://www.quantamagazine.org/moon-duchin-on-fair-voting-and-random-walks-20200407/`.

[41] Moon Duchin and Bridget Eileen Tenner. Discrete geometry for electoral geography. *arXiv preprint arXiv:1808.05860*, 2018.

[42] Martin E Dyer and Alan M Frieze. On the complexity of partitioning graphs into connected subgraphs. *Discrete Applied Mathematics*, 10(2):139–153, 1985.

[43] Jonathan Eckstein. Personal communication, 2020.

[44] Andreas Eisenblätter. The semidefinite relaxation of the k-partition polytope is strong. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 273–290. Springer, 2002.

[45] Yuri Faenza. Personal communication, 2021.

[46] Yuri Faenza and Volker Kaibel. Extended formulations for packing and partitioning orbitopes. *Mathematics of Operations Research*, 34(3):686–697, 2009.

[47] Carlos E Ferreira, Alexander Martin, C Carvalho de Souza, Robert Weismantel, and Laurence A Wolsey. Formulations and valid inequalities for the node capacitated graph partitioning problem. *Mathematical Programming*, 74(3):247–266, 1996.

[48] Carlos Eduardo Ferreira, Alexander Martin, C Carvalho de Souza, Robert Weismantel, and Laurence A Wolsey. The node capacitated graph partitioning problem: a computational study. *Mathematical Programming*, 81(2):229–256, 1998.

[49] Benjamin Fifield, Michael Higgins, Kosuke Imai, and Alexander Tarr. A new automated redistricting simulator using Markov chain Monte Carlo. *Work. Pap., Princeton Univ., Princeton, NJ*, 2015.

[50] Matteo Fischetti, Markus Leitner, Ivana Ljubić, Martin Luipersbeck, Michele Monaci, Max Resch, Domenico Salvagnin, and Markus Sinnl. Thinning out Steiner trees: a node-based model for uniform edge costs. *Mathematical Programming Computation*, 9(2):203–229, 2017.

[51] Matteo Fischetti, Andrea Lodi, and Domenico Salvagnin. Just MIP it! In *Matheuristics*, pages 39–70. Springer, 2009.

[52] Fabio Furini, Ivana Ljubić, Enrico Malaguti, and Paolo Paronuzzi. On integer and bilevel formulations for the k-vertex cut problem. *Mathematical Programming Computation*, 12(2):133–164, 2020.

[53] Fabio Furini, Ivana Ljubić, Enrico Malaguti, and Paolo Paronuzzi. Casting light on the hidden bilevel combinatorial structure of the capacitated vertex separator problem. *Operations Research*, 2021. To appear.

[54] Robert S Garfinkel and George L Nemhauser. Optimal political districting by implicit enumeration techniques. *Management Science*, 16(8):B–495, 1970.

[55] Ambros Gleixner, Gregor Hendel, Gerald Gamrath, Tobias Achterberg, Michael Bastubbe, Timo Berthold, Philipp Christophel, Kati Jarck, Thorsten Koch, Jeff Linderoth, et al. MIPLIB 2017: data-driven compilation of the 6th mixed-integer programming library. *Mathematical Programming Computation*, 2021. To appear.

[56] Sebastian Goderbauer and Jeff Winandy. Political districting problem: Literature review and discussion with regard to federal elections in Germany, 2018.

[57] Michel X Goemans and Young-Soo Myung. A catalog of Steiner tree formulations. *Networks*, 23(1):19–28, 1993.

[58] Olivier Goldschmidt and Dorit S Hochbaum. A polynomial algorithm for the k-cut problem for fixed k. *Mathematics of Operations Research*, 19(1):24–37, 1994.

[59] Ram Gopalan, Steven O Kimbrough, Frederic H Murphy, and Nicholas Quintus. The Philadelphia districting contest: Designing territories for city council based upon the 2010 census. *Interfaces*, 43(5):477–489, 2013.

[60] Bernard Grofman. Criteria for districting: A social science perspective. *UCLA L. Rev.*, 33:77, 1985.

[61] Martin Grötschel and Yoshiko Wakabayashi. A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45(1):59–96, 1989.

[62] Martin Grötschel and Yoshiko Wakabayashi. Facets of the clique partitioning polytope. *Mathematical Programming*, 47(1):367–387, 1990.

[63] Diansheng Guo and Hai Jin. iRedistrict: Geovisual analytics for redistricting optimization. *Journal of Visual Languages & Computing*, 22(4):279–289, 2011.

[64] Wes Gurnee and David B Shmoys. Fairmandering: A column generation heuristic for fairness-optimized political districting. *arXiv preprint arXiv:2103.11469*, 2021.

[65] Miguel Ángel Gutiérrez-Andrade, Eric Alfredo Rincón-García, Sergio Gerardo de-los Cobos-Silva, Pedro Lara-Velázquez, Roman Anselmo Mora-Gutiérrez, and Antonin Ponsich. Simulated annealing and artificial bee colony for the redistricting process in Mexico. *INFORMS Journal on Applied Analytics*, 49(3):189–200, 2019.

[66] Pierre Hansen and Brigitte Jaumard. Cluster analysis and mathematical programming. *Mathematical Programming*, 79(1):191–215, 1997.

[67] David Hartvigsen. The planar multiterminal cut problem. *Discrete Applied Mathematics*, 85(3):203–222, 1998.

[68] J Gerald Hebert, Martina E Vandenberg, and Paul Smith. *The Realist's Guide to Redistricting: Avoiding the Legal Pitfalls*. American Bar Association, 2010.

[69] SW Hess, JB Weaver, HJ Siegfeldt, JN Whelan, and PA Zitlau. Nonpartisan political redistricting by computer. *Operations Research*, 13(6):998–1006, 1965.

[70] Cyrus Hettle, Shixiang Zhu, Swati Gupta, and Yao Xie. Balanced districting on grid graphs with provable compactness and contiguity. *arXiv preprint arXiv:2102.05028*, 2021.

[71] Mehran Hojati. Optimal political districting. *Computers & Operations Research*, 23(12):1147–1161, 1996.

[72] Christopher Hojny, Imke Joormann, Hendrik Lüthen, and Martin Schmidt. Mixed-integer programming techniques for the connected max-k-cut problem. *Mathematical Programming Computation*, 13(1):75–132, 2021.

[73] Christopher Hojny and Marc E Pfetsch. Polytopes associated with symmetry handling. *Mathematical Programming*, 175(1-2):197–240, 2019.

[74] Ellis L Johnson, Anuj Mehrotra, and George L Nemhauser. Min-cut clustering. *Mathematical Programming*, 62(1-3):133–151, 1993.

[75] Volker Kaibel, Matthias Peinhardt, and Marc E Pfetsch. Orbitopal fixing. *Discrete Optimization*, 8(4):595–610, 2011.

[76] Volker Kaibel and Marc Pfetsch. Packing and partitioning orbitopes. *Mathematical Programming*, 114(1):1–36, 2008.

[77] Haim Kaplan and Yahav Nussbaum. Maximum flow in directed planar graphs with vertex capacities. *Algorithmica*, 61(1):174–189, 2011.

[78] Aaron Kaufman, Gary King, and Mayya Komisarchik. How to measure legislative district compactness if you only know it when you see it. *American Journal of Political Science*, Forthcoming.

[79] Myung Kim and Ningchuan Xiao. Contiguity-based optimization models for political redistricting problems. *International Journal of Applied Geospatial Research (IJAGR)*, 8(4):1–18, 2017.

[80] Myung Jin Kim. Give-and-take heuristic model to political redistricting problems. *Spatial Information Research*, 27:539–552, 2019.

[81] Douglas M King, Sheldon H Jacobson, and Edward C Sewell. Efficient geograph contiguity and hole algorithms for geographic zoning and dynamic plane graph partitioning. *Mathematical Programming*, 149(1-2):425–457, 2015.

[82] Douglas M King, Sheldon H Jacobson, and Edward C Sewell. The geo-graph in practice: creating United States congressional districts from census blocks. *Computational Optimization and Applications*, 69(1):25–49, 2018.

[83] Douglas M King, Sheldon H Jacobson, Edward C Sewell, and Wendy K Tam Cho. Geo-graphs: an efficient model for enforcing contiguity and hole constraints in planar graph partitioning. *Operations Research*, 60(5):1213–1228, 2012.

[84] Martine Labbé and F Aykut Özsoy. Size-constrained graph partitioning polytopes. *Discrete Mathematics*, 310(24):3473–3493, 2010.

[85] Jakub Lacki, Yahav Nussbaum, Piotr Sankowski, and Christian Wulff-Nilsen. Single source–all sinks max flows in planar digraphs. In *Foundations of Computer Science (focs), 2012 IEEE 53rd Annual Symposium on*, pages 599–608. IEEE, 2012.

[86] Euiwoong Lee. Partitioning a graph into small pieces with applications to path transversal. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1546–1558. SIAM, 2017.

[87] Harry A Levin and Sorelle A Friedler. Automated congressional redistricting. *Journal of Experimental Algorithmics (JEA)*, 24(1):1–10, 2019.

[88] Justin Levitt. *A citizen's guide to redistricting*. Brennan Center for Justice at New York University School of Law, 2010.

[89] John M Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

[90] Yan Y Liu, Wendy K Tam Cho, and Shaowen Wang. PEAR: a massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and Evolutionary Computation*, 30:78–92, 2016.

[91] Thomas L Magnanti and Laurence A Wolsey. Optimal trees. *Handbooks in Operations Research and Management Science*, 7:503–615, 1995.

[92] François Margot. Pruning by isomorphism in branch-and-cut. *Mathematical Programming*, 94(1):71–90, 2002.

[93] François Margot. Exploiting orbits in symmetric ILP. *Mathematical Programming*, 98(1-3):3–21, 2003.

[94] François Margot. Small covering designs by branch-and-cut. *Mathematical Programming*, 94(2-3):207–220, 2003.

[95] François Margot. Symmetric ILP: Coloring and small integers. *Discrete Optimization*, 4(1):40–62, 2007.

[96] François Margot. Symmetry in integer linear programming. In *50 Years of Integer Programming 1958-2008*, pages 647–686. Springer, 2010.

[97] Anuj Mehrotra, Ellis L Johnson, and George L Nemhauser. An optimization based heuristic for political districting. *Management Science*, 44(8):1100–1114, 1998.

[98] Anuj Mehrotra and Michael A Trick. Cliques and clustering: A combinatorial approach. *Operations Research Letters*, 22(1):1–12, 1998.

[99] MGGG. GerryChain 0.2.12., 2021. `https://gerrychain.readthedocs.io/en/latest/`.

[100] Stacy Miller. The problem of redistricting: the use of centroidal Voronoi diagrams to build unbiased congressional districts. *Senior project, Whitman College*, 2007.

[101] Flávio K Miyazawa, Phablo FS Moura, Matheus J Ota, and Yoshiko Wakabayashi. Partitioning a graph into balanced connected classes: formulations, separation and experiments. *European Journal of Operational Research*, 2021. To appear.

[102] Stuart S Nagel. Simplified bipartisan computer redistricting. *Stan. L. Rev.*, 17:863, 1964.

[103] Richard G Niemi, Bernard Grofman, Carl Carlucci, and Thomas Hofeller. Measuring compactness and the role of a compactness standard in a test for partisan and racial gerrymandering. *The Journal of Politics*, 52(4):1155–1181, 1990.

[104] Johannes Oehrlein and Jan-Henrik Haunert. A cutting-plane method for contiguity-constrained spatial aggregation. *Journal of Spatial Information Science*, 2017(15):89–120, 2017.

[105] Brian Olson. Impartial automatic redistricting. `https://bdistricting.com/2010/`, 2019. Accessed: 2019-06-21.

[106] Maarten Oosten, Jeroen HGC Rutten, and Frits CR Spieksma. The clique partitioning problem: facets and patching facets. *Networks: An International Journal*, 38(4):209–226, 2001.

[107] Maarten Oosten, Jeroen HGC Rutten, and Frits CR Spieksma. Disconnecting graphs by removing vertices: a polyhedral approach. *Statistica Neerlandica*, 61(1):35–60, 2007.

[108] James Ostrowski, Miguel F Anjos, and Anthony Vannelli. Modified orbital branching for structured symmetry with an application to unit commitment. *Mathematical Programming*, 150(1):99–129, 2015.

[109] James Ostrowski, Jeff Linderoth, Fabrizio Rossi, and Stefano Smriglio. Orbital branching. *Mathematical Programming*, 126(1):147–178, 2011.

[110] Marc Pfetsch. Personal communication, 2021.

[111] Marc E Pfetsch and Thomas Rehn. A computational comparison of symmetry handling methods for mixed integer programs. *Mathematical Programming Computation*, 11(1):37–93, 2019.

[112] Richard H Pildes and Richard G Niemi. Expressive harms, "bizarre districts," and voting rights: Evaluating election-district appearances after Shaw v. Reno. *Michigan Law Review*, 92(3):483–587, 1993.

[113] Daniel D Polsby and Robert D Popper. The third criterion: Compactness as a procedural safeguard against partisan gerrymandering. *Yale L. & Pol'y Rev.*, 9:301, 1991.

[114] Daniel D Polsby and Robert D Popper. Ugly: An inquiry into the problem of racial gerrymandering under the Voting Rights Act. *Mich. L. Rev.*, 92:652, 1993.

[115] Daniel Rehfeldt, Henriette Franz, and Thorsten Koch. Optimal connected subgraphs: Formulations and algorithms. Technical Report 20-23, ZIB, Takustr. 7, 14195 Berlin, 2020.

[116] Ernest C Reock. A note: Measuring compactness as a requirement of legislative apportionment. *Midwest Journal of Political Science*, 5(1):70–74, 1961.

[117] Federica Ricca, Andrea Scozzari, and Bruno Simeone. Weighted Voronoi region algorithms for political districting. *Mathematical and Computer Modelling*, 48(9-10):1468–1477, 2008.

[118] Federica Ricca, Andrea Scozzari, and Bruno Simeone. Political districting: from classical models to recent approaches. *Annals of Operations Research*, 204(1):271–299, 2013.

[119] Federica Ricca and Bruno Simeone. Local search algorithms for political districting. *European Journal of Operational Research*, 189(3):1409–1426, 2008.

[120] Hosseinali Salemi and Austin Buchanan. Parsimonious formulations for low-diameter clusters. *Mathematical Programming Computation*, 12(3):493–528, 2020.

[121] Stephan Schwartz. An overview of graph covering and partitioning. Technical Report 20-24, ZIB, Takustr. 7, 14195 Berlin, 2020.

[122] Joseph E Schwartzberg. Reapportionment, gerrymanders, and the notion of compactness. *Minn. L. Rev.*, 50:443, 1965.

[123] Siqian Shen, J Cole Smith, and Roshan Goli. Exact interdiction models and algorithms for disconnecting networks via node deletions. *Discrete Optimization*, 9(3):172–188, 2012.

[124] Takeshi Shirabe. A model of contiguity for spatial unit allocation. *Geographical Analysis*, 37(1):2–16, 2005.

[125] Takeshi Shirabe. Districting modeling with exact contiguity constraints. *Environment and Planning B: Planning and Design*, 36(6):1053–1066, 2009.

[126] Michael M Sørensen. Facet-defining inequalities for the simple graph partitioning polytope. *Discrete Optimization*, 4(2):221–231, 2007.

[127] Bhushan Suwal, Matthew Sun, and Parker Rule. mggg/GerryChainJulia: v0.1.2, October 2020. `https://doi.org/10.5281/zenodo.4111000`.

[128] Lukas Svec, Sam Burden, and Aaron Dilley. Applying Voronoi diagrams to the redistricting problem. *The UMAP Journal*, 28(3):313–329, 2007.

[129] Rahul Swamy, Douglas M. King, and Sheldon H. Jacobson. A case for transparency in the design of political districts, 2019. Working paper.

[130] Rahul Swamy, Douglas M. King, and Sheldon H. Jacobson. Multi-objective optimization for political districting: a scalable multilevel approach, 2019. Working paper.

[131] Hamidreza Validi, Austin Buchanan, and Eugene Lykhovyd. Imposing contiguity constraints in political districting models. *Operations Research*, 2021. To appear.

[132] William Vickrey. On the prevention of gerrymandering. *Political Science Quarterly*, 76(1):105–110, 1961.

[133] Yiming Wang, Austin Buchanan, and Sergiy Butenko. On imposing connectivity constraints in integer programs. *Mathematical Programming*, 166(1-2):241–271, 2017.

[134] Justin C Williams Jr. Political redistricting: a review. *Papers in Regional Science*, 74(1):13–40, 1995.

[135] Mingyu Xiao. Simple and improved parameterized algorithms for multiterminal cuts. *Theory of Computing Systems*, 46(4):723–736, 2010.

[136] Mingyu Xiao. Linear kernels for separating a graph into components of bounded size. *Journal of Computer and System Sciences*, 88:260–270, 2017.

[137] H Peyton Young. Measuring the compactness of legislative districts. *Legislative Studies Quarterly*, 13(1):105–115, 1988.

Appendix A – More Contiguity Constraints

Below, we give a single-commodity flow (SCF) model for the Hess base model (2). It is inspired by the SCF model of Hojny et al. [72] which uses a flow variable f_{ij} for each arc (i, j) from the directed graph $H = (V, A)$.

$$f(\delta^-(j)) - f(\delta^+(j)) = 1 - \sum_{i \in V} x_{ij} \qquad \forall j \in V \qquad (15a)$$

(Hess-SCF)

$$f(\delta^-(j)) \leq M(1 - x_{jj}) \qquad \forall j \in V \qquad (15b)$$

$$f_{ij} + f_{ji} \leq M(1 - y_e) \qquad \forall e = \{i, j\} \in E \qquad (15c)$$

$$f_{ij} \geq 0 \qquad \forall (i, j) \in A. \qquad (15d)$$

Constraints (15a) enforce that if vertex j is *not* assigned to itself, then it should consume one unit of flow; otherwise, j should send out one unit of flow for every *other* vertex i that is assigned to j. Constraints (15b) disallow flow into vertices j that are assigned to themselves. Constraints (15c) disallow flow across cut edges. In our experiments, we set M by equation (4).

Next, we give a multi-commodity flow (SHIR) model for the Labeling base model (1) that is inspired by Shirabe [104, 124, 125]. It uses binary root variables r_{ij} that equal one when vertex $i \in V$ is assigned to district $j \in [k]$. It also uses a flow variable f^j_{uv} for each district ("commodity") $j \in [k]$ and each arc $(u, v) \in A$. We also define a variable g^j_i for each vertex $i \in V$ and district $j \in [k]$ indicating how much flow of commodity j is generated at (root) vertex $i \in V$.

$$\sum_{i \in V} r_{ij} = 1 \qquad \forall j \in [k] \qquad (16a)$$

$$r_{ij} \leq x_{ij} \qquad \forall i \in V,\ \forall j \in [k] \qquad (16b)$$

$$g^j_i \leq (M + 1) r_{ij} \qquad \forall i \in V,\ \forall j \in [k] \qquad (16c)$$

(Label-SHIR)

$$f^j(\delta^-(i)) - f^j(\delta^+(i)) = x_{ij} - g^j_i \qquad \forall i \in V,\ \forall j \in [k] \qquad (16d)$$

$$f^j(\delta^-(i)) \leq M(x_{ij} - r_{ij}) \qquad \forall i \in V,\ \forall j \in [k] \qquad (16e)$$

$$g^j_i \geq 0 \qquad \forall i \in V,\ \forall j \in [k] \qquad (16f)$$

$$f^j_{uv} \geq 0 \qquad \forall (u, v) \in A,\ \forall j \in [k] \qquad (16g)$$

$$r_{ij} \in \{0, 1\} \qquad \forall i \in V,\ \forall j \in [k]. \qquad (16h)$$

Constraints (16a) force each district j to have one root. Constraints (16b) state that each vertex i cannot root a district j to which it does not belong. Constraints (16c) ensure that flow of commodity $j \in [k]$ is only generated at its root. Constraints (16d) force vertex i to consume one unit of commodity j flow if it is assigned to district j. Constraints (16e) disallow flow into roots of districts; they also disallow flow of commodity $j \in [k]$ into vertex i if it is not assigned to district j. Again, we set M by equation (4).

Appendix B – More Experiments with Model (12)

Table 15: Results for model (12) for $q \in \{k, 2k\}$ bins under one-minute and one-hour time-limits (TL). Note that the times differ slightly from those in Table 4 due to noise between runs.

			60-second time-limit				3600-second time-limit			
			k bins		$2k$ bins		k bins		$2k$ bins	
state	n	k	$\|B\|$	time	$\|B\|$	time	$\|B\|$	time	$\|B\|$	time
ME	16	2	13	0.03	13	0.05	13	0.02	13	0.06
NM	33	3	28	0.17	28	0.53	28	0.20	28	0.48
ID	44	2	41	0.05	41	0.20	41	0.03	41	0.19
WV	55	3	48	3.72	48	2.02	48	3.69	48	1.95
LA	64	6	53	30.41	53	36.53	53	30.43	53	36.61
AL	67	7	52	TL	52	TL	52	232.42	52	1,110.05
AR	75	4	64	23.78	64	22.82	64	23.74	64	22.88
OK	77	5	64	20.63	64	TL	64	20.73	64	152.76
MS	82	4	69	TL	69	TL	69	113.11	69	1,020.68
NE	93	3	86	2.31	86	2.87	86	2.36	86	2.91
IA	99	4	85	TL	85	TL	85	141.93	85	1,098.42
KS	105	4	95	12.55	95	16.23	95	12.43	95	16.01
NH	295	2	283	7.44	283	TL	283	7.42	283	353.50
ID	298	2	291	2.55	292	38.10	291	2.59	292	37.84
ME	358	2	349	3.31	349	36.12	349	3.30	349	35.99
WV	484	3	467	TL	464	TL	467	TL	467	TL
NM	499	3	485	TL	482	TL	485	TL	485	TL
NE	532	3	513	TL	510	TL	513	TL	513	TL
UT	588	4	556	TL	552	TL	564	TL	557	TL
MS	664	4	634	TL	633	TL	637	TL	637	TL
AR	686	4	653	TL	653	TL	657	TL	656	TL
NV	687	4	653	TL	651	TL	656	TL	656	TL