OPERATIONS RESEARCH
Vol. 60, No. 5, September–October 2012, pp. 1213–1228
ISSN 0030-364X (print) | ISSN 1526-5463 (online)

http://dx.doi.org/10.1287/opre.1120.1083

Geo-Graphs: An Efficient Model for Enforcing Contiguity and Hole Constraints in Planar Graph Partitioning

Douglas M. King
Department of Industrial and Enterprise Systems Engineering, University of Illinois, Urbana, Illinois 61801,
dmking@illinois.edu

Sheldon H. Jacobson
Department of Computer Science, University of Illinois, Urbana, Illinois 61801,
shj@illinois.edu

Edward C. Sewell
Department of Mathematics and Statistics, Southern Illinois University Edwardsville, Edwardsville, Illinois 62026,
esewell@siue.edu

Wendy K. Tam Cho
Department of Political Science and Statistics, National Center for Supercomputing Applications,
University of Illinois, Urbana, Illinois 61801, wendycho@illinois.edu

Political districting is an intractable problem with significant ramifications for political representation. Districts often are required to satisfy some legal constraints, but these typically are not very restrictive, allowing decision makers to influence the composition of these districts without violating relevant laws. For example, while districts must often comprise a single contiguous area, a vast collection of acceptable solutions (i.e., sets of districts) remains. Choosing the best set of districts from this collection can be treated as a (planar) graph partitioning problem. When districts must be contiguous, successfully solving this problem requires an efficient computational method for evaluating contiguity constraints; common methods for assessing contiguity can require significant computation as the problem size grows. This paper introduces the *geo-graph*, a new graph model that ameliorates the computational burdens associated with enforcing contiguity constraints in planar graph partitioning when each vertex corresponds to a particular region of the plane. Through planar graph duality, the geo-graph provides a scale-invariant method for enforcing contiguity constraints in local search. Furthermore, geo-graphs allow district holes (which typically are considered undesirable) to be rigorously and efficiently integrated into the partitioning process.

Subject classifications: graphs/theory: plane graph partitioning; government/elections: political districting; graphs/heuristics: local search.
Area of review: Optimization.
History: Received December 2010; revisions received December 2011, February 2012; accepted March 2012. Published online in *Articles in Advance* October 9, 2012.

1. Introduction

The boundaries of United States Congressional Districts must be redrawn every 10 years in response to population shifts measured by the national census. This process is an example of *political districting*, which is the process of dividing a geographic region into a set of districts, often for the purpose of electing political representatives. While relevant laws constrain some facets of the districting process, these constraints typically are not very restrictive, and therefore decision makers are able to exert influence over district composition. Because these districts have significant ramifications on political representation, many groups (e.g., political parties, public interest groups) have conflicting views on how these districts should be created. For example, political parties might favor districts that are most likely to elect their candidates, while public interest groups might prefer districts where elections will be more competitive. Butler and Cain (1992) discuss the underlying principles that guide redistricting, while di Cortona et al. (1999) provide numerical measures of some common redistricting objectives. Because different groups would like to quickly identify districting options that are most in line with their objectives, they can view districting as a constrained optimization problem. Bozkaya et al. (2011) present a case where the districts produced by an optimization approach were adopted by the city of Edmonton, Canada, demonstrating the practical applicability of optimization-based redistricting. Due to the diversity of districting objectives, a general districting algorithm cannot be tailored to a specific type of objective.

The redistricting process typically is discrete in nature, as the entire region is first discretized into a large but finite set of small areas (called *basic units* or simply *units*) that are then combined to form districts. One common requirement is that these districts be contiguous. The discrete nature of such problems, coupled with the presence of contiguity constraints, makes a *graph* the natural choice to model this problem. A graph, $G = (V, E)$, comprises a set of vertices, V, and a set of edges, E, with each edge joining two vertices. In districting applications, vertices represent the basic units, and edges join pairs of adjacent units (i.e., units with a common boundary). Under this construction, the graph G is planar (i.e., it can be embedded in the plane such that no two edges intersect other than at a common endpoint). A district is considered contiguous if the subgraph induced by its vertices is connected. Grouping the units into districts becomes a *graph partitioning* problem that seeks to optimize the objective specified by the designer.

Graph partitioning is an intractable discrete optimization problem. While many types of graph partitioning problems exist, many that appear relatively simple have been shown to be *NP*-complete in the general case (Garey et al. 1976). Heuristics such as local search have been adopted to generate good (though suboptimal) solutions in a reasonable time frame. Local search begins with and iteratively perturbs an initial solution until a stopping criterion is met; under this paradigm, search time depends on (1) the number of iterations until the stopping criterion is met, and (2) the amount of time required to execute each iteration. Both of these facets are influenced by a solution's neighborhood (i.e., the set of solutions that can be generated by perturbing the current solution); a large neighborhood might permit exploration of a large set of solutions in each iteration, but evaluating the quality of these solutions requires additional computation. Computation can be reduced when the underlying architecture of the problem informs the choice of neighborhood.

In graph partitioning, the neighborhood can be adapted to take advantage of the structure of the graph and its constraints. One key feature of districting is that each vertex corresponds to a particular region of the plane (i.e., the associated unit). This relationship suggests a second graph whose edges draw the boundaries of these regions and whose vertices are the points where three or more boundary segments intersect. Both graphs are related through planar duality, and can be simultaneously embedded in the same plane (Whitney 1932). Though only the original graph will be partitioned, geometry from the second graph can illuminate decisions made about the original graph when they are superimposed. Outside of districting, this correspondence between vertices and regions of the plane can also be observed in other geographic zoning applications (e.g., Ricca and Simeone 2008, D'Amico et al. 2002), as well as image segmentation (e.g., Wang 1998, Shi and Malik 2000). This paper will adopt the more specific term "zone" in place of the more general "partite set" or the more restrictive "district" to signify that the vertices in each partite set correspond to a particular zone of the plane. Similarly, "zoning" will be used in place of "districting."

This paper presents a new model, termed the *geo-graph*, that formalizes the dual relationship between the two graphs present in zoning applications and demonstrates how this connection can be exploited to more quickly assess two types of zoning constraints: contiguity constraints and hole constraints. When local search perturbs a set of zones (typically by transferring a single unit to a different zone), the contiguity of the new zones must be evaluated. One simple approach is to execute a graph search on each of the affected zones, rejecting the perturbation if any search fails to visit all of its vertices. Verifying the contiguity of zone i in this approach requires $O(|V_i|)$ time, where V_i is the set of units in zone i. Such extensive computation is cumbersome when applied to large problems such as creating United States Congressional Districts, where the average number of census blocks per district in a state varied from 9,495 (Hawaii) to 45,685 (New Mexico) after the 2000 Census (United States Census Bureau 2000, 2011).

By contrast, this paper will show how a geo-graph avoids large-scale searches and assesses contiguity of the perturbed zones by examining only the units appearing on the boundary of the transferred unit. This analysis requires $O(m(G)|R(v)|)$ time when transferring unit v, where $m(G)$ is the number of zones and $R(v)$ is the set of units that appear on the boundary of v. Hole constraints, common in geographic zoning, prohibit any zone from being an enclave of another zone. Although cited as important in geographic zoning, they have yet to be rigorously integrated into mathematical approaches to zoning (Tavares-Pereira et al. 2007). A geo-graph identifies and enumerates the holes of a zone in $O(m(G)^2)$ time by maintaining an auxiliary graph that requires $O(|R(v)|)$ time to update after local search transfers unit v. For both types of constraints, computation depends on two factors: the size of $R(v)$ and the number of zones, neither of which necessarily increases with the number of units in the region. For example, consider a region that is to be divided into five zones. If the units that compose this region are arranged in a rectangular grid, then $|R(v)| \leqslant 8$ regardless of its dimensions. In this example, a geo-graph allows the designer to select a finer grid with a much larger set of zoning options without affecting the time complexity of contiguity assessments. Such scale invariance is critical when solving large zoning problems.

By design, the geo-graph algorithms described in this paper determine whether a particular local search transition is *feasible* under contiguity or hole constraints, not whether the transition is *beneficial* to the goals of the designer; the latter decision is left entirely to the designer. Therefore, the geo-graph can be applied to zoning problems without regard to their objective (e.g., maximum competitiveness, maximum population balance) or mechanism for choosing a feasible transition at each local search iteration (e.g., strict

descent, tabu search, simulated annealing). This modularity allows the geo-graph model to be incorporated into a large number of zoning problems; it also means that using a geo-graph will not affect the quality of solutions generated by local search, but will allow these solutions to be generated more quickly.

The paper is organized as follows. Section 2 surveys existing work in graph partitioning and zoning that motivates the development of geo-graphs. Section 3 formalizes the structure and notation used in the geo-graph model, whose compatibility with practical districting problems is discussed in §4. Section 5 proposes a computationally efficient method for identifying zone holes within the geo-graph paradigm, while §6 reveals the formal relationship between contiguity and hole constraints and develops computationally efficient methods for evaluating zone contiguity in local search. Section 7 discusses the particular case of creating United States Congressional Districts in Kansas to demonstrate how using the geo-graph model reduces the amount of computation required to assess contiguity when compared with simple search methods in practical districting applications. Finally, §8 discusses these results and draws conclusions about how geo-graphs can improve existing local search approaches to partitioning planar graphs with many vertices. Proofs of all theorems, lemmas, and corollaries are included in an online companion. An electronic companion to this paper is available as part of the online version at http://dx.doi.org/10.1287/opre.1120.1083.

2. Background

Graph partitioning seeks to divide the vertex set of a graph into two or more subsets; this division is frequently presented as an optimization problem, where the partition is generated according to domain-specific constraints and objectives. Most geographic zoning problems consider contiguity and balance constraints, where a set of zones is considered balanced if it distributes some quantity evenly among its zones. For example, balance in political districting requires districts to be approximately equipopulous, while sales districts must balance workload among their salesforce. Political districting objectives are influenced by factors such as population demographics, voting patterns, geography, district shape, and integrity of communities of interest (Butler and Cain 1992), depending on the goals of the designer. The political districting process of assigning n units to k zones can be characterized by the mathematical program

$$\begin{aligned}
\max_{\gamma\in\Gamma}\ & obj(\gamma),\\
\text{s.t.}\ & L_p \leqslant Pop_j(\gamma) \leqslant U_p \quad \forall\, j = 1, 2, \ldots, k,\\
& Conn_j(\gamma) = 1 \quad \forall\, j = 1, 2, \ldots, k,\\
& Empty_j(\gamma) = 0 \quad \forall\, j = 1, 2, \ldots, k,
\end{aligned}$$

where γ represents the assignment of the units to their zones, with Γ representing the set of all such assignments (i.e., $|\Gamma| = k^n$), $obj(\gamma)$ is a maximization objective reflecting the goals of the designer, $Pop_j(\gamma)$ is the population of zone j under the assignments in γ with upper and lower bounds U_p and L_p defining the feasible range for the balance constraint, $Conn_j(\gamma)$ is an indicator function that equals one when zone j is contiguous and zero otherwise, and $Empty_j(\gamma)$ is an indicator function that equals one when zone j is empty (i.e., no units have been assigned to it) and zero otherwise; note that $Empty_j(\gamma) = 0$ is automatically satisfied if $L_p > 0$. This program can be realized as a binary integer program when the state variable γ is represented as an $n \times n$ matrix of binary variables, $\gamma_{i,j}$, that take a value of one if unit i is assigned to a zone centered at unit j, and zero otherwise, where these units are taken from the set of units, V. This formulation is adapted from Drexl and Haase (1999); for $i \in V$, element $\gamma_{i,i}$ is equal to one if and only if there is a zone centered at vertex i, otherwise it is zero. Note that each zone contains exactly one "center"; other than being contained within the zone, this center does not necessarily have additional meaning with regard to the shape of the zone. The resulting binary integer program is

$$\max \quad obj(\gamma), \tag{1}$$

$$\text{s.t.} \quad L_p\gamma_{j,j} \leqslant \sum_{i\in V} p_i\gamma_{i,j} \leqslant U_p\gamma_{j,j} \quad \forall\, j \in V, \tag{2}$$

$$\sum_{j\in V} \gamma_{j,j} = k, \tag{3}$$

$$\sum_{j\in V} \gamma_{i,j} = 1 \quad \forall\, i \in V, \tag{4}$$

$$\gamma_{i,j} \leqslant \gamma_{j,j} \quad \forall\, i, j \in V, \tag{5}$$

$$\sum_{v_1\in\bigcup_{s\in S} N(s)-S} \gamma_{v_1,j} - \sum_{v_2\in S} \gamma_{v_2,j} \geqslant 1 - |S| \quad \forall\, j \in V \text{ and } S \subseteq (V - N(j) - j), \tag{6}$$

$$\gamma_{i,j} \in \{0, 1\} \quad \forall\, i, j \in V, \tag{7}$$

where for each unit $i \in V$, p_i is its population and $N(i)$ is the set of units adjacent to it. The constraint in Equation (2) enforces population balance, while Equation (3) requires that there are exactly k zones, Equation (4) ensures that each unit is assigned to exactly one zone, Equation (5) mandates that each unit is assigned to a zone whose center has been established, Equation (6) enforces zone contiguity, and Equation (7) defines the binary nature of $\gamma_{i,j}$. Enforcing the constraints in Equation (6) requires a number of inequalities that increases exponentially with the number of units, making it intractable for large problems. A fluid flow approach to contiguity yields a mixed-integer program formulation that avoids this exponential increase by adding continuous decision variables measuring flow volume, although this formulation is also intractable in large problems (Shirabe 2005, 2009). While many variations of graph partitioning exist beyond political districting, most are similarly intractable. For example, the *Minimum Cut into Equal-Sized Subsets* problem asks whether

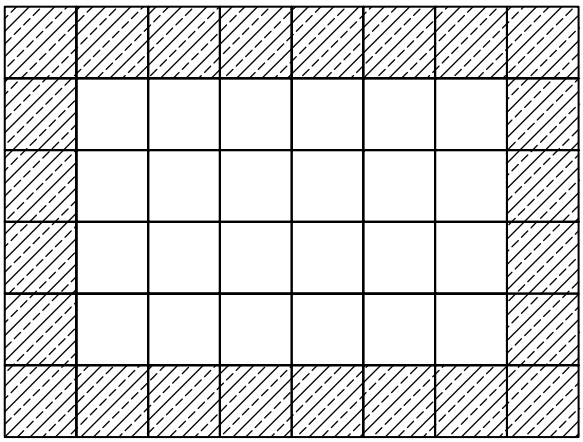
a graph, $G \quad V \; E$, with two marked vertices, $v_1 \;\; v_2 \quad V$, and a positive integer, $W \qquad$, can have its vertex set divided into two subsets, V_1 and V_2 (with $V_1 \quad V_2 \qquad$ and $V_1 \quad V_2 \quad V$), such that $v_1 \quad V_1$, $v_2 \quad V_2$, $V_1 \qquad V_2$, and $xy \quad E \;\; x \quad V_1 \;\; y \quad V_2 \qquad W$ (i.e., the "cut" refers to edges that have endpoints in different sets). Although this problems lacks the contiguity requirement inherent to geographic zoning problems, Minimum Cut Into Equal-Sized Subsets has been shown to be *NP*-complete (Garey et al. 1976).

Heuristic approaches have been developed to mitigate the intractability of these graph partitioning problems. The most well-known graph partitioning heuristic for minimum-cut problems is the Kernighan-Lin algorithm (Kernighan and Lin 1970), which initially partitions the vertices into two sets of equal cardinality and iteratively improves this partition by determining a series of vertex trades, where one vertex from each set is transferred to the other, thereby maintaining an equal number of vertices in each set. Fiduccia and Mattheyses (1982) refine this algorithm by replacing vertex trades with single vertex transfers; balance is enforced by bounding the cardinality of each set rather than enforcing strict equality. Both algorithms reduce computational effort by recycling objective computations across iterations. When a single vertex is transferred, as in the Fiduccia-Mattheyses algorithm, the minimum-cut objective is affected only by edges that change from "cut" to "uncut" or vice versa; these edges will be a subset of those incident to the transferred vertex, which typically make up a small portion of the entire edge set.

Graph partitioning problems become more complex when one imposes contiguity constraints, such as in many geographic zoning problems (e.g., Hansen et al. 2003, Nygreen 1988, Wang 1998, Becker et al. 1998). Conceptually, zone contiguity requires that for every pair of points in the interior of a zone, one can draw a curve between them that passes through only the interior of that zone. While enforcing zone contiguity with inequality constraints is intractable, graph contiguity can be assessed through a graph search algorithm; beginning at one vertex, this search determines whether all other vertices can be visited by traversing the edges of the graph. Although attractive in its simplicity, the time complexity of evaluating graph contiguity for a planar graph is linear in its number of vertices, and the associated computation can be substantial when the graph is large.

If the current zones are contiguous, and local search transfers one vertex, $v \quad V$, from its current set (the *sending set*) to another set (the *receiving set*) in each iteration, then all zones but these two will remain contiguous after this transition. The receiving set will remain contiguous if and only if at least one of the edges incident to the transferred vertex has a vertex of the receiving set as its other endpoint; this condition can be checked in $O \;\; N \; v \quad$ time. Contiguity of the sending set is more difficult to assess; Ricca and Simeone (2008) propose checking its contiguity by marking the neighbors of v that are in the sending set, temporarily removing v from the sending set, and beginning a graph search of the sending set at one of the marked vertices. The sending set remains contiguous if and only if this search visits all of the marked vertices. While this search requires $O \;\; V_i \quad$ time, where $V_i \subset V$ is the sending set, it may visit relatively few vertices when the sending set remains contiguous. However, when this set becomes discontiguous the search will visit every vertex in one of its remaining components before arriving at this conclusion. Furthermore, there are cases when the search might need to visit every vertex in the sending set even when the sending set does not become discontiguous. For example, consider the hatched zone in Figure 1. Each vertex in the hatched zone has two hatched neighbors; if this vertex is removed, the only path between these neighbors passes through every vertex in the hatched zone. As the number of vertices in the graph increases, the average size of each partite set will grow accordingly, and the computational cost of a search-based approach that might need to visit nearly all the vertices in the sending set will limit the effectiveness of partitioning algorithms in large zoning problems. While this search can be made more efficient for zoning applications using dynamic graph models for planar graphs such as those proposed by Frigioni and Italiano (2000) and Eppstein et al. (1992), which require polylogarithmic time, rather than linear time required by traditional search algorithms, the time required to assess zone contiguity will still increase with the size of the graph. Furthermore, using these models might require complex data structures and algorithms that might make them unattractive to practitioners. Macmillan (2001) discusses how contiguity of the sending set can be assessed by examining the "switching points" that occur on the boundary of the transferred unit, noting the zone holes can influence this analysis; the geo-graph model rigorously integrates zone holes into contiguity analysis of the sending set, developing a formal graph theoretic framework that analyzes contiguity

Figure 1. Example 6 8 grid region with two zones.

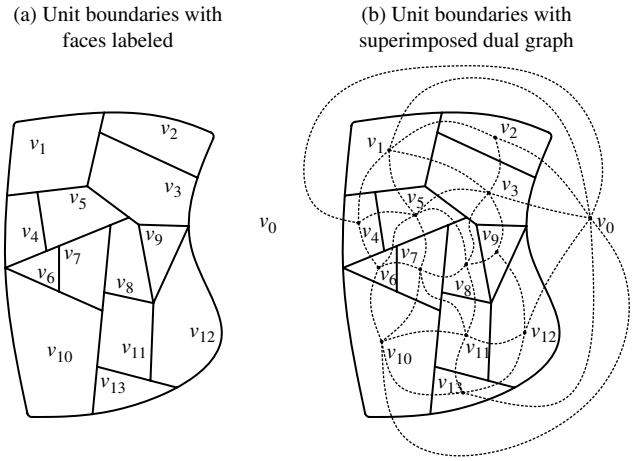


by only examining units that share a boundary with the transferred unit.

The concept of zone *holes* is similar to that of zone contiguity in that holes are defined by curves drawn in the interior of a zone. A hole occurs when one can draw a simple closed curve in the interior of a zone, such that the entire area enclosed by the curve is not part of that zone; in geographic zoning, these holes typically are considered undesirable (e.g., di Cortona et al. 1999, Ricca 2004, Tavares-Pereira et al. 2007). While holes can be easily identified by a person viewing the zones as drawn on a map, identifying them computationally is more challenging and has not been rigorously incorporated into graph partitioning algorithms (Tavares-Pereira et al. 2007). A compactness objective tends to penalize zones that violate contiguity and hole constraints (di Cortona et al. 1999) and hence, often leads to zones that do not violate these constraints. In political districting, compactness typically is encouraged as a surrogate for more germane objectives, such as preserving the integrity of natural communities of interest (Butler and Cain 1992), and impedes overt tampering with districts for political gain, commonly called *gerrymandering*. Therefore, maximum compactness objectives frequently appear in optimization approaches to political districting, where many numerical indices of a district's compactness have been proposed (e.g., di Cortona et al. 1999, Bozkaya et al. 2003). However, compactness is not typically a districting requirement from a legal standpoint, and its inclusion can bias a local search algorithm, leading the search away from optimal solutions that satisfy contiguity and hole constraints but whose districts are not compact. While compactness might encourage desirable district attributes in many cases, optimizing these attributes directly eliminates the bias that compactness injects into the optimization process. Efficient methods for directly evaluating contiguity and hole constraints are needed.

To identify zone holes, one must know how the units are arranged in the plane. In a geo-graph with n units, each unit corresponds to one of n finite subregions of the plane. Drawing the unit boundaries in the plane divides the plane into n 1 areas, with the additional area corresponding to the infinite area outside the unit boundaries; it is assumed that this infinite area is contiguous. This drawing can be interpreted as an embedded planar graph whose dual can be embedded on the same plane (Whitney 1932). This embedding of the dual graph is constructed as follows: one vertex of the dual graph is placed in the interior of the area of each unit, and an edge is drawn between two vertices of the dual graph for each unbroken segment of border that they share in the original graph (excluding isolated points); since there could be several such segments, the dual might be a multigraph. Each dual edge connects its endpoints by traveling through the interiors of their corresponding areas of the original graph, crossing at a single point on their common border (except the border's endpoints). For example, Figures 2(a) and 3(a) depict two example regions

Figure 2. Construction of the dual for a general region.

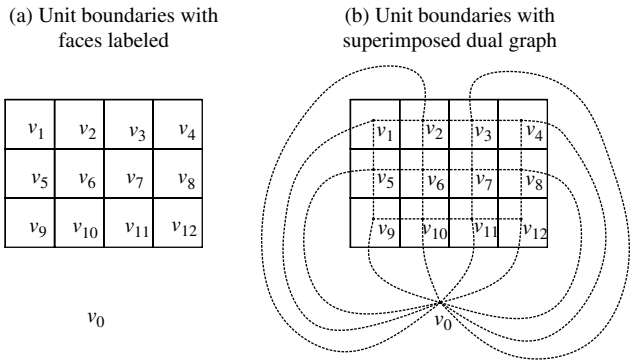


whose dual graphs are superimposed in Figures 2(b) and 3(b), respectively. The dual graph summarizes the entities (i.e., subregions) and adjacencies that are used when creating zones. By this construction, both graphs are simultaneously embedded in the same plane. Their superimposition can provide additional information about the structure of the graph and will be useful when assessing contiguity constraints and holes in graph partitioning.

3. Geo-Graph Definitions and Terminology

Geo-graphs provide a modeling framework for planar graph partitioning problems when each vertex corresponds to a particular finite area (i.e., unit) of the plane. This section defines the geo-graph, focusing on how the boundaries of these units dictate its structure. The geo-graph, G V E B z , augments the usual vertex and edge sets with two functions: a boundary function, B, and a zoning function, z. The zoning function z partitions the vertex set by assigning each vertex to a single zone. The number of such zones is defined by m G , and therefore z is a function from V to the set of zones, defined as M G 1 2 m G .

Figure 3. Construction of the dual of a 3 4 grid-shaped region.



Furthermore, for a set of zones $K \subseteq M(G)$, let the set of vertices in these zones be defined as $V(K) \equiv \{v \in V: z(v) \in K\}$; if $k \in M(G)$, then the notation $V(k)$ will be used rather than $V(\{k\})$. The contiguity of a zone is equivalent to the connectivity of the subgraph induced by the vertices in that zone. To this end, geo-graph G is called *zone-connected* if for every zone $k \in M(G)$ the subgraph induced by $V(k)$ is connected.

The boundary function defines the relationship between the vertices in V and their corresponding units by associating each vertex, $v \in V$, with the simple closed curve, $B(v)$, that constitutes the boundary of that unit. Two assumptions are made about these curves. First, drawing these curves on the plane divides the plane into exactly $|V|+1$ regions; therefore, any point on the plane that does not fall on or inside one of the $B(v)$ curves must belong to the single infinite area outside of the geographic region. Second, $B(v)$ is a single curve for each $v \in V$, which prevents any unit from comprising discontiguous pieces or containing any holes. If either assumption is violated, these violations typically can be reconciled by either adding units to fill in empty areas (when the curves divide the plane into more than $|V|+1$ regions) or merging two or more basic units (when $B(v)$ consists of more that one curve). In general, these assumptions are not very restrictive and will be suitable for many applications.

While the vertices in V correspond to $|V|$ of the regions of the plane defined by B, the region corresponding to the infinite area outside of these units is considered through an additional dummy vertex, v_0, with the function B extended such that $B(v_0)$ denotes the simple closed curve that separates the infinite area outside the region from the units. Although v_0 serves a different purpose than the vertices in V (e.g., it does not represent a unit and will not be a part of any zone) and is therefore excluded from V, there are cases when explicit consideration of v_0 will be useful. For this reason, define the augmented set of vertices as $V_0 = V \cup \{v_0\}$; furthermore, v_0 is always assigned to a dummy zone, $z(v_0) = 0$, and the augmented set of zones is denoted by $M_0(G) = \{0, 1, 2, \ldots, m(G)\}$. With the addition of v_0, the geo-graph G is exactly the dual of the curves prescribed by the boundary function, B.

Through duality, the boundary function implicitly defines the edge set, E, for the geo-graph, and the *neighborhood* of vertex $v \in V_0$ is defined in the standard way as $N(v) \equiv \{x \in V_0: vx \in E\}$. However, these adjacencies reflect only vertices that share segments on their common border. Although units sharing isolated points on their common border should not be considered adjacent for the purposes of contiguity, knowing what pairs of units share such isolated points provides valuable information about the arrangement of units in the plane. Therefore, the *augmented neighborhood* of v is defined as $R(v) \equiv \{x \in V_0: B(x) \cap B(v) \neq \varnothing\}$. By design, $N(v) \subseteq R(v)$, as the augmented neighborhood provides a less restrictive definition of adjacency. For example, vertex v_7 in Figure 3(b) has $N(v_7) = \{v_3, v_6, v_8, v_{11}\}$ and $R(v_7) = \{v_2, v_3, v_4, v_6, v_8, v_{10}, v_{11}, v_{12}\}$, and $N(v_7)$ is a strict subset of $R(v_7)$, whereas vertex v_7 in Figure 2(b) has $N(v_7) = R(v_7) = \{v_5, v_6, v_8, v_{10}, v_{11}\}$. In general, both neighborhoods are symmetric; for any two vertices $x, y \in V$ in any geo-graph, both of the following hold: $x \in N(y)$ if and only if $y \in N(x)$, and $x \in R(y)$ if and only if $y \in R(x)$. Furthermore, the set of *zone neighbors* of vertex v in zone $k \in M_0(G)$ is defined by $N_k(v) \equiv N(v) \cap V(k)$, and v is called *neighbor-connected* in zone k when $N_k(v)$ is contained in a single component of the subgraph induced by $R(v) \cap V(k)$.

To support both neighborhoods, the definition of a *path* must be extended. A *path*, $P = (p_1, p_2, \ldots, p_j)$, is defined as a sequence of vertices where $p_{i-1} \in N(p_i)$ for $2 \leqslant i \leqslant j$ and no vertex is repeated. If this sequence has $p_1 \in N(p_j)$, then P is called a *cycle* (typically denoted by C); each edge can appear only once in a cycle, and hence a two-vertex cycle must contain two distinct edges between these vertices. A *strand* is similarly defined as a sequence of vertices, $S = (s_1, s_2, \ldots, s_j)$, where $s_{i-1} \in R(s_i)$ for $2 \leqslant i \leqslant j$ and no vertex is repeated; if $s_1 \in R(s_j)$, then S is called a *closed strand* (typically denoted by C^S). Following from the restriction on cycles, a closed strand can be composed of only two vertices when there are at least two distinct segments (or isolated points) on the shared boundary of their units. When discussing the visitation order of the vertices in a path or strand, the forward order (i.e., from p_1 to p_j or s_1 to s_j) is assumed by convention.

While $R(v)$ enumerates the set of vertices whose units share at least a single point on $B(v)$, the curve specified by the boundary function also determines the order in which these vertices are encountered while traveling around $B(v)$. As $B(v)$ is a simple closed curve, cutting this curve at two distinct points yields two simple curves that connect those two points but do not otherwise intersect. Each *perimeter* of vertex $v \in V$ lists the sequence of units encountered while traveling along one of these curves. Because the same vertex could be encountered more than once over one of these curves, the perimeter might not conform to the definition of a path. Rather, each perimeter is a *walk*, which is defined in the same way as a path but might repeat vertices and edges (West 2001). These perimeters can be constructed as follows.

Definition 1. For any vertex $v \in V$, choose any $x_1, x_2 \in R(v)$; choosing $x_1 = x_2$ is permissible when $B(v) \cap B(x_1)$ is not a single segment or isolated point. Define an x_1, x_2*-perimeter on* v as an x_1, x_2-walk that is constructed as follows:

1. Let a_1 be the point on $B(v) \cap B(x_1)$ where an edge between these two vertices crosses the shared border (or an isolated point on the shared border), and let a_2 be the point on $B(v) \cap B(x_2)$ where an edge between these two vertices crosses the shared border (or an isolated point on the shared border). If $x_1 = x_2$, then a_1 and a_2 must be distinct and cannot both appear on the same segment of $B(v) \cap B(x_1)$. Let $L \subseteq B(v)$ be a curve that connects a_1 and a_2 on $B(v)$. Let $t = a_1$ be the current position on L, $W = (x_1)$ be the

sequence of vertices encountered between a_1 and t along L, and r x_1 be the most recently visited vertex on P.

2. Advance t along L until it reaches either a_2, or a point p where one or more boundary curves intersect L. If it reaches a_2, return the walk W. Otherwise, the curves intersecting L at p must be B r B w_1 , B w_1 B w_2 B w_{m-1} B w_m for some sequence of vertices w_1 w_2 w_m R v . Append this sequence of vertices to the end of the walk W, update t p and r w_m, and repeat Step 2.

If v_0 W, then W is called *broken*.

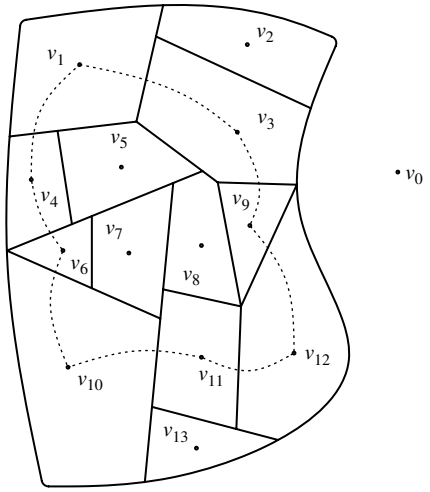
To demonstrate, consider the two v_8, v_{13}-perimeters on v_{11} in Figure 2(b). The perimeter that travels clockwise around the boundary of v_{11} is v_8 v_9 v_{12} v_{13} , while the perimeter that travels counter-clockwise around the boundary is v_8 v_7 v_{10} v_{13} . Neither perimeter is broken. In contrast, the v_1, v_{10}-perimeters on v_5 in Figure 3(b) are v_1 v_2 v_6 v_{10} traveling clockwise and v_1 v_0 v_9 v_{10} traveling counter-clockwise; the counter-clockwise perimeter is broken.

Duality gives the edges of a geo-graph a specific embedding in the plane, and hence each cycle is associated with a simple closed curve composed of its edges. This curve separates the plane into two regions: the region inside the closed curve and the infinite region outside. A similar closed curve can be defined for each closed strand; as for a cycle, this curve is composed of curves drawn between neighboring vertices in the strand. This curve corresponds either to an edge connecting these two vertices or to a curve segment that passes through the interiors of the units associated with the vertices and one of the isolated points on their shared border. If the closed curve associated with a closed strand does not intersect itself, then this is a simple closed curve, and the closed strand is called *tangle-free*; a cycle is one example of a tangle-free closed strand. By dividing the plane into two regions, the simple closed curve associated with a tangle-free closed strand can classify all vertices in the graph based on which region the vertex occupies. Definition 2 defines the set of vertices that are *internal* and *external* to a tangle-free closed strand.

Definition 2. For any tangle-free closed strand C^S V and any vertex v V C^S, v is said to be *internal* to C^S if and only if vertex v is located inside the simple closed curve associated with C^S. Define *Int* C^S as the set of vertices that are internal to this curve, and *Ext* C^S V_0 *Int* C^S C^S as the set of vertices that are *external* to the curve. By definition, v_0 *Ext* C^S , as v_0 represents an infinite area that cannot be contained within the finite area internal to the curve.

Figure 4 depicts the edges of a single cycle on the vertices of the region shown in Figure 2. The cycle has the vertex sequence C v_1 v_3 v_9 v_{12} v_{11} v_{10} v_6 v_4 . Based on the simple closed curve drawn by the edges of C, its internal vertices are *Int* C v_5 v_7 v_8 , and its external vertices are *Ext* C v_0 v_2 v_{13} . It should be noted that, though they are related, the terms *internal* and *interior* are not equivalent. Consider any tangle-free closed strand: while the interior of the area enclosed by its curve contains an infinite number of points, the internal vertices of the strand occur at a finite set of those points. Based on the definition of internal vertices, a zone will be called *surrounded* when its vertices lie completely inside the curve drawn by a cycle of vertices from another zone.

Figure 4. The edges of a cycle C v_1 v_3 v_9 v_{12} v_{11} v_{10} v_6 v_4 , which has *Int* C v_5 v_7 v_8 and *Ext* C v_0 v_2 v_{13} .



Definition 3. A zone k M G is called *surrounded* if there exists a cycle of vertices, C V j , for some j M G k, such that V k *Int* C . In this case it is said that C *surrounds* zone k; more generally, zone j is said to surround zone k.

This definition demonstrates the relationship between zone holes and surrounded zones. When a zone is surrounded, it must appear in a hole in the surrounding zone, since the curve associated with the surrounding cycle only passes through the interior of the surrounding zone, and the entire area of the surrounded zone lies inside that curve. Furthermore, due to the restrictions on the boundary function, B, holes and surrounded zones are interchangeable; any hole must be filled with one or more zones. One obstacle remains in declaring equivalence between holes and surrounded zones: there are only finitely many simple closed curves associated with cycles, while there are an infinite number of simple closed curves that can be drawn in the interior of a zone. Later in this paper, Lemma 6 will show that a hole exists if and only if it lies within the simple closed curve associated with a cycle, demonstrating that holes and surrounded zones are equivalent.

For a zone k M G , let k M G k be the set of zones surrounded by zone k. For each zone in j k , there must be a closed curve through the units of zone k

such that all the units of zone j are enclosed by this curve. If another zone $j' \in \Pi(k) - j$ must also be enclosed by this curve (for any such curve), then j and j' are said to be in the same *pocket* of zone k. Definition 4 shows how the set of surrounded zones $\Pi(k)$ can be decomposed into classes of pockets.

Definition 4. Let $G = (V, E, B, z)$ be a zone-connected geo-graph. Define $\Pi(k) \subseteq M(G) - k$ as the set of zones surrounded by zone $k \in M(G)$. Furthermore, define the *pocket set* of zone k as $\{\Pi_1(k), \Pi_2(k), \ldots, \Pi_{\pi(k)}(k)\}$, such that $\bigcup_{i=1}^{\pi(k)} \Pi_i(k) = \Pi(k)$ and $\Pi_{i_1}(k) \cap \Pi_{i_2}(k) = \varnothing$ for $1 \leqslant i_1 < i_2 \leqslant \pi(k)$, where for each $1 \leqslant j \leqslant \pi(k)$ and $p, q \in V(\Pi_j(k))$, there is a p, q-strand, S, such that $S \subseteq V(\Pi_j(k))$.

As Theorem 1 will show, knowing the pocket set of a zone allows zone contiguity in local search to be assessed by examining the vertices in $R(v)$ when vertex v is being transferred. Lemma 9 shows how the pockets of a zone can be efficiently identified by examining an auxiliary graph, $H(G)$, defined in Definition 5, that summarizes the inter-zone adjacencies of geo-graph G.

Definition 5. Define $H(G) = (V', E')$ as an auxiliary graph to zone-connected geo-graph $G = (V, E, B, z)$. This auxiliary graph has $V' = M_0(G)$, such that vertex k in $H(G)$ corresponds to zone k in G. For any two vertices, $k_1, k_2 \in V'$, there is an edge $k_1k_2 \in E'$ if and only if there exist vertices, $x_1, x_2 \in V$ such that $z(x_1) = k_1$, $z(x_2) = k_2$, and $x_1 \in R(x_2)$.

The structure of $H(G)$ will evolve as local search transfers units between the zones, and therefore the pocket set of each zone will also evolve. Section 5 discusses how the structure of the auxiliary graph can be updated in $O(|R(v)|)$ time after vertex v migrates to a different zone in local search.

4. Compatibility with Practical Political Districting

By emphasizing zone contiguity in large graph partitioning problems, the geo-graph model is specifically tailored to solve practical political districting problems. This section demonstrates its suitability by examining the construction of United States Congressional Districts.

Every two years, voters in each district elect a representative to the United States House of Representatives. Under the "one person, one vote" principle, the populations of these districts must be approximately equal. Every 10 years, a national census measures population shifts in the country; district boundaries must be redrawn to accommodate these shifts. First, population shifts at the national level are used to apportion seats among the states based on state populations; a state might gain or lose seats if its population rises or falls relative to the other states. Second, if a state is apportioned multiple representatives, it must partition its area among the same number of contiguous districts. Each district must be contained entirely within an individual state, and hence each state's redistricting process is independent of the other states. While there are rare occasions when other constraints arise, such as legally mandated "majority minority" districts that prevent a geographically concentrated minority populations from being diluted among several districts, these two constraints (contiguity and equal population) are the primary factors in political redistricting.

The input data required to use the geo-graph model are available through government sources. To assess contiguity, the geo-graph requires both the standard and augmented neighbor sets for each unit (i.e., $N(v)$ and $R(v)$ for each $v \in V_0$). Using *geographic information system* (GIS) software (e.g., ArcGIS), these data can be extracted from GIS shapefiles published by the United States government (United States Census Bureau 2010b); while some census blocks might not have boundaries that are simple closed curves, these violations are rare, can be identified using GIS software, and can be addressed with only superficial impact to the solution space. Population data for enforcing population balance constraints, as well as demographic, voting pattern, and shape data used to compute objective values within the optimization process are also available from publicly accessible databases (e.g., United States Census Bureau 2010a, Minnesota Population Center 2004, Missouri Census Data Center 2010).

The United States Census Bureau collects and publishes population data for census blocks, which are the smallest geographic units considered in the census. In geographic terms, census blocks are typically bounded by entities such as roads, rivers, and other natural frontiers. These data can be clustered to form more coarse levels of detail (e.g., block groups, census tracts), but census blocks represent the finest level of detail for population data and therefore provide the most flexibility when designing districts. Each state contains a very large number of census blocks: between 17,483 (Delaware) and 675,062 (Texas) during the 2000 Census (United States Census Bureau 2011). By contrast, each state is divided into a relatively small number of districts; at 53 districts, California had the most after the 2000 Census, while seven states (Alaska, Delaware, Montana, North Dakota, South Dakota, Vermont, and Wyoming) each encompassed a single district (United States Census Bureau 2000). In the remaining 43 states, the number of census blocks per district ranges from 9,495 (Hawaii) to 45,685 (New Mexico), with an average of 22,516 census blocks per district. In all cases, the number of census blocks exceeds the number of districts by a factor of more than 9,000. The size of the solution space in each districting problem is astronomical. Among the states apportioned more than one seat, Hawaii has both the fewest seats (two) and the fewest census blocks (18,990), yet there are $2^{18,989} - 1 > 10^{5,700}$ ways to generate these districts when contiguity and balance constraints are neglected. While limiting choices to

only contiguous and balanced zones significantly reduces this number, the set of feasible solutions remains enormous. Furthermore, assessing contiguity in local search using a simple search method might need to visit every unit in the examined district. For problems on the scale of United States Congressional Districts, this approach will prevent local search from exploring the solution space quickly and, hence, a more efficient method to assess zone contiguity is needed.

5. Identifying Surrounded Zones

While the basic structure and properties of a geo-graph were discussed in §3, this section further develops these properties and demonstrates how they can be used to solve large zoning problems efficiently. For example, while defining surrounded zones might be interesting from a theoretical perspective, the results presented in this section show that surrounded zones (1) are equivalent to holes and (2) can be identified in a computationally efficient way. By contrast, the only avenue afforded by the definition of surrounded zones is to enumerate the cycles composed of vertices from a single zone and determine whether another zone is internal to each. Such a combinatorially exhaustive approach would require a Herculean degree of computation. The results presented in this section bridge the gap between theoretical novelty and computational practicality.

Although this section deals primarily with surrounded zones, the focus on contiguity constraints is not abandoned, as these concepts are naturally intertwined. For example, when local search removes vertex v from zone $z(v)$, the remainder of the zone is contiguous if and only if the subgraph induced by $V(z(v)) - v$ is connected. Lemma 1 shows that assessing the connectivity of this graph is equivalent to identifying a cycle in $V(z(v))$ for each pair of vertices in $N_{z(v)}(v)$. While finding such a cycle for each pair of vertices can also be computationally expensive, the equivalence of these two conditions will be needed in the proofs of several later theorems and lemmas.

Lemma 1. *Let $G = (V, E, B, z)$ be a zone-connected geo-graph. For any $v \in V$, the subgraph induced by $V(z(v)) - v$ is connected if and only if for every pair of vertices $x, y \in N_{z(v)}(v)$, there is a cycle $C \subseteq V(z(v))$ in which x, v, y appear consecutively.*

When a vertex appears on a tangle-free closed strand, its boundary curve and the curve associated with the strand will intersect. Both are simple closed curves, with two intersection points defined where the strand curve enters and exits the boundary. If the boundary is cut at these two points, the two simple curves that remain are exactly those associated with the two perimeters on the vertex; one can show that one of these perimeter curves lies inside the simple closed curve associated with the tangle-free closed strand and one lies outside (either perimeter curve may also intersect, but not cross, the curve associated with the strand). Lemma 2 extends the perimeter curve classifications to the vertices that are visited along these perimeter curves.

Lemma 2. *Let $G = (V, E, B, z)$ be a geo-graph, $C^S \subseteq V$ be a tangle-free closed strand on which $x, v, y \in V$ appear consecutively, and $W_1, W_2 \subseteq R(v)$ be the two x, y-perimeters on v.*

A. *For some $j \in \{1, 2\}$, $W_j \cap N(v) \subseteq C^S \cup Int(C^S)$ and $W_{3-j} \cap N(v) \subseteq C^S \cup Ext(C^S)$.*

B. *If $C = C^S$ is a cycle, then for some $j \in \{1, 2\}$, $W_j \subseteq C \cup Int(C)$ and $W_{3-j} \subseteq C \cup Ext(C)$. Furthermore, W_j is unbroken and therefore is a x, y-walk on $R(v) \cap (C \cup Int(C))$.*

While this lemma allows one to classify vertices as internal and external to a tangle-free closed strand, Lemma 3 shows how classifying one vertex as either internal or external to such a strand allows other vertices to be classified in a methodical and straightforward way and allows these classifications to be extended to entire zones. For example, Lemmas 3A and 3B allow the classification of one vertex to be extended to other vertices in either its neighborhood or augmented neighborhood, depending on whether C^S is a cycle.

Lemma 3. *Let $G = (V, E, B, z)$ be a geo-graph, with $C^S \subset V$ being any tangle-free closed strand, and $x \in V - C^S$. The following properties hold*:

A. *For every $y \in N(x) - C^S$, $y \in Int(C^S)$ if and only if $x \in Int(C^S)$.*

B. *If $C = C^S$ is a cycle, then for every $y \in R(x) - C$, $y \in Int(C)$ if and only if $x \in Int(C)$.*

C. *If $C = C^S$ is a cycle and $B(x) \cap B(v_0) \neq \varnothing$, then $x \notin Int(C)$.*

D. *If $y \in V - C^S$ such that $x \in Int(C^S)$ and $y \in Ext(C^S)$, then each x, y-path, P, has $P \cap C^S \neq \varnothing$.*

E. *If $C = C^S$ is a cycle and $y \in V - C$ such that $x \in Int(C)$ and $y \in Ext(C)$, then each x, y-strand, S, has $S \cap C \neq \varnothing$.*

F. *If G is zone-connected with $z(x) = i$ for some $i \in M(G)$, $C = C^S$ is a cycle, and $C \subseteq V(j)$ for some $j \in M(G) - i$, then zone i is surrounded by C if and only if $x \in Int(C)$.*

While conceptually simple, these lemmas are powerful; for example, Lemma 3B allows every vertex in $V - C$ to be classified by classifying only one vertex in each component of the subgraph induced by $V - C$, rather than classifying each vertex individually. However, a method to actually classify any particular vertex as internal or external to a cycle is not clear without drawing the curve traced by the cycle, determining the area it encloses, and checking whether the vertex is contained in that area; carrying out this analysis could require significant computation.

Rather than create a cycle and then ask whether a particular vertex is internal or external to it, one can designate a set of vertices and ask whether it is possible to construct

a cycle to which these vertices are internal. For specific types of vertex subsets, Lemma 4 can answer this question. If a tangle-free closed strand is sought (rather than a cycle), Lemma 5 is used.

LEMMA 4. *Let $G = V, E, B, z$ be a geo-graph. If there is a subset of vertices, $U \subseteq V$, such that for every u_1, $u_2 \in U$, there is a u_1, u_2-strand on the vertices of U, and $B(u) \subseteq B(v_0)$ for all $u \in U$, then there is a cycle, C $_{u \in U} R(u) \subseteq U$, such that $U \subseteq Int(C)$.*

LEMMA 5. *Let $G = V, E, B, z$ be a geo-graph. If there is a subset of vertices, $T \subseteq V$, such that for every t_1 t_2 $\in T$, there is a t_1, t_2-path on the vertices of T, and $B(t) \subseteq B(v_0)$ for all $t \in T$, then there is a tangle-free closed strand, C^S $_{t \in T} N(t) \subseteq T$, such that $T \subseteq Int(C^S)$.*

The ability to construct a surrounding cycle in Lemma 4 shows one way to identify surrounded zones. Lemma 6 demonstrates that it is possible to draw a simple closed curve through the interior of a zone such that another zone lies entirely inside this curve if and only if the first zone contains a cycle that surrounds the second zone. Therefore, only curves traced by cycles need to be considered when identifying holes. This result is similar to using edges to verify contiguity; although edges represent only finitely many of the infinite number of simple curves that can be drawn between two units, these other curves do not need to be considered when assessing contiguity.

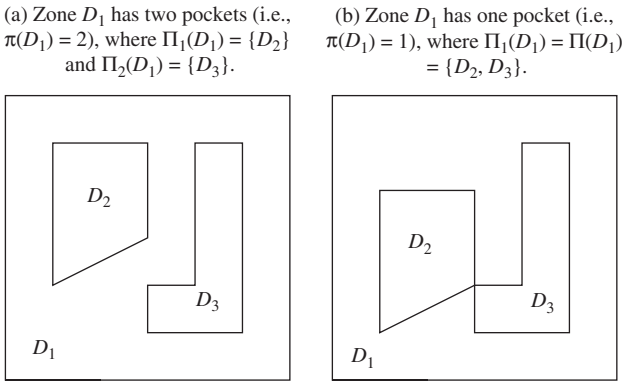
LEMMA 6. *Let $G = V, E, B, z$ be a zone-connected geo-graph with distinct zones i_1 i_2 $\in M(G)$. There is a cycle, $C \subseteq V(i_1)$ such that $V(i_2) \subseteq Int(C)$ if and only if every vertex in $V(i_2)$ is contained in the area bounded by a simple closed curve, L, that passes through the interior of the area associated with zone i_1 (i.e., the union of the areas of the units associated with vertices in $V(i_1)$).*

While the previous lemmas in this section refer to single vertices or single zones that are surrounded by a cycle, Lemma 3B can show that if two augmented neighbors are in different zones, then if one vertex is internal to a cycle, the other will also be internal; if the graph is zone-connected, Lemma 3F can show that this cycle either surrounds both zones or neither zone. This observation leads directly to the classification of surrounded zones into pockets, as in Definition 4, which are applied in Lemma 7.

LEMMA 7. *Let $G = V, E, B, z$ be a zone-connected geo-graph. For any zone $j \in M(G)$, let $C \subseteq V(j)$ be a cycle in zone j. For any $1 \le i \le j$, if there is a vertex $x \in V_i(j)$ such that $x \in Int(C)$, then $V_i(j) \subseteq Int(C)$.*

To demonstrate how pockets are identified, consider the two examples in Figure 5. In both cases, there are three zones: D_1, D_2, and D_3, where zones D_2 and D_3 are surrounded by zone D_1 (i.e., $D_1 = D_2, D_3$). In Figure 5(a), each surrounded zone is in a separate pocket of zone D_1, since there is no shared point on their common borders (i.e., $D_1 = 2$, $_1(D_1) = D_2$, and $_2(D_1) = D_3$).

Figure 5. Identification of pockets in a zoning problem with zones D_1, D_2, D_3, where $D_1 = D_2, D_3$.



From the perspective of holes, one can draw a simple closed curve through the interior of zone D_1 that encloses only one of the surrounded zones. By contrast, the two surrounded zones share a single point on their border in Figure 5(b), and therefore both zones are in the same pocket of D_1 (i.e., $D_1 = 1$ and $_1(D_1) = D_1 = D_2, D_3$). Furthermore, no simple closed curve drawn through the interior of zone D_1 can enclose only one of these zones; it must enclose both zones or neither zone.

Following from the example in Figure 5(b), two surrounded zones must be in the same pocket of a surrounding zone if their is at least one common point on their shared boundary. Any point on the boundary of a zone must also be a point on the boundary of one of the units in that zone, which implies that these pockets can be defined in terms of the augmented neighborhoods of the units in geo-graph G. Definition 5 shows that an auxiliary graph to G, defined as $H(G)$, can efficiently translate these intervertex relationships into their corresponding interzone relationships.

In addition to concisely summarizing classes of surrounded zones, the auxiliary graph also allows these surrounded zones to be identified in a computationally efficient way. Lemma 8 shows that one can identify whether zone $i \in M(G)$ surrounds zone $j \in M(G)$, i by removing vertex i from $H(G)$ and beginning a graph search at vertex j; if this search does not visit vertex 0, then zone i surrounds zone j. Lemma 9 extends this result to show that the pocket set of zone i can be identified by removing i from $H(G)$ and enumerating the components of the remaining graph.

LEMMA 8. *Let $G = V, E, B, z$ be a zone-connected geo-graph, with associated auxiliary graph $H(G) = V, E$. For any pair of zones, i, $j \in V$, $\neq 0$, there is a cycle, $C \subseteq V(i)$ in G, such that C surrounds zone j if and only if there is no j, 0-path in $H(G)$ that does not pass through i.*

LEMMA 9. *Let $G = V, E, B, z$ be a zone-connected geo-graph, with associated auxiliary graph $H(G) = V, E$. For any zone $k \in M(G)$, let $T_1, T_2, \ldots, T_m$ be the components of the subgraph induced by $V - k$ in $H(G)$, with*

$0 \in T_m$. *Zone* k *has* $\pi(k) = m - 1$ *pockets, with pocket set defined by* $\Pi_i(k) = T_i$ *for* $1 \leqslant i \leqslant \pi(k)$.

Lemma 9 suggests a computationally efficient method for identifying surrounded zones. The auxiliary graph $H(G)$ has $m(G)+1$ vertices and potentially $\binom{m(G)+1}{2}$ edges, and hence, a simple graph search algorithm (e.g., depth-first search) can enumerate these components in $O(m(G)^2)$ time. In zoning applications the number of zones is typically much smaller than the number of units, and hence, surrounded zones (and the pockets in which they reside) can be identified with relatively little computation. Furthermore, the complexity of these computations does not grow with the number of vertices in G as long as the same number of zones are being created; pocket identification is scale invariant to how finely or coarsely the region is divided into units.

One obstacle in implementing the auxiliary graph, $H(G)$, is that its edges are determined by the zoning function z, and hence, edges may be added or removed as units migrate from one zone to another during local search. To allow $H(G)$ to evolve as units are transferred, the auxiliary graph can be implemented using a square matrix with dimension $|V'|$, such that element (i, j) of this matrix is equal to $|\{(x, y) \in V_0 \times V_0 \colon x \in V(i), y \in V(j), x \in R(y)\}|$, the number of pairs of augmented neighbors such that one is in each of zones i and j. Then, edge $ij \in E'$ if and only if element (i, j) of this matrix is greater than zero. This implementation is attractive from a local search perspective, as this matrix can be updated efficiently each time a vertex is moved to a new zone, requiring only one addition and one subtraction for each vertex in its augmented neighborhood. When vertex v is transferred to a new zone, the entries of the matrix can be updated in $O(|R(v)|)$ time. As with identifying pocket sets, such computations will be scale invariant to the number of units being considered, as long as the size of the augmented neighborhoods remains consistent at different scales. For example, if the units are arranged in a grid pattern, then $|R(v)| \leqslant 8$, regardless of the grid's dimensions. The change in the average size of $R(v)$ for a practical political districting problem will be discussed in §7.

6. Main Results

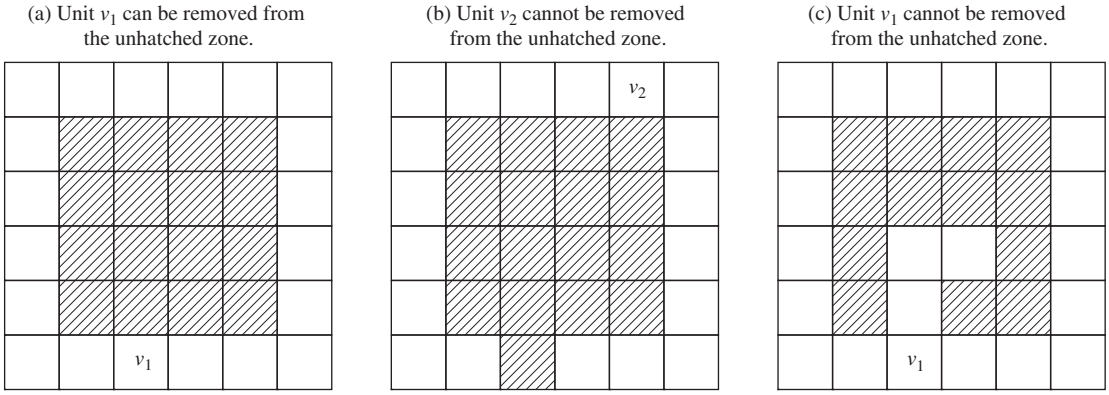
When local search partitions a geo-graph, $G = (V, E, B, z)$, into contiguous zones, enforcing contiguity can be computationally expensive, even when each iteration transfers only one vertex $v \in V$ from its current zone to a different zone. This expense comes when evaluating the contiguity of zone $z(v)$, which will lose vertex v. A natural question is: under what conditions will this approach become computationally expensive, and can these expenses be controlled? The results presented in this section demonstrate how these expenses can be controlled by considering surrounded zones to the extent that they become scale invariant to the number of vertices in the graph.

To see how surrounded zones can inform contiguity assessments, note that the graph search method used by Ricca and Simeone (2008) verifies contiguity by finding an x, y-path on $V(z(v)) - v$ for each pair of vertices $x, y \in N_{z(v)}(v)$. Appending v to each path produces a cycle $C \subseteq V(z(v))$ on which x, v, y appear consecutively. Therefore, $Int(C)$ must either contain (1) only vertices of $V(z(v)) - v$, in which case there is an x, y-path on $R(v) \cap V(z(v)) \subseteq V(z(v)) - v$ by Lemma 2B; or (2) at least one vertex that is not in $V(z(v))$, in which case C surrounds a zone by Lemma 3F. In the first case, a path can be found that passes through only vertices in $R(v)$; by considering surrounded zones, Theorem 1 shows that both cases can be assessed by examining vertices in $R(v)$.

THEOREM 1. *Let* $G = (V, E, B, z)$ *be a zone-connected geo-graph with* $v \in V$. *The subgraph induced by* $V(z(v)) - v$ *is connected if and only if* $N_{z(v)}(v)$ *is contained in a single component of the subgraph induced by* $R(v) \cap V(\Pi(z(v)) \cup \{z(v)\})$, *and for every* $1 \leqslant j \leqslant \pi(z(v))$, $N_{z(v)}(v)$ *is contained in a single component of the subgraph induced by* $R(v) \cap V(M_0(G) - \Pi_j(z(v)))$.

To see how these conditions verify contiguity, consider the example region whose units are depicted in Figure 6(a). The units in the unhatched zone compose a single cycle that surrounds the hatched zone. Suppose that vertex $v_1 \in V$ is to be transferred from the unhatched zone to the hatched zone. For the first condition of Theorem 1, $R(v_1) \cap V(\Pi(z(v_1)) \cup \{z(v_1)\})$ contains five vertices in $R(v_1)$: three hatched units in the row above v_1 and the two unhatched units to either side. The subgraph induced by these vertices has a single component, which therefore must contain both unhatched neighbors of v_1; conceptually, this condition allows graph search to pass through the surrounded zone rather than traveling around it. For the second condition, $\pi(z(v_1)) = 1$, and $R(v_1) \cap V(M_0(G) - \Pi_1(z(v_1)))$ contains both unhatched neighbors of v_1 and the vertex v_0. As in the first condition, the subgraph induced by these vertices has a single component, which includes both vertices in $N_{z(v_1)}(v_1)$. Having satisfied these two conditions, Theorem 1 can conclude that the unhatched zone will remain contiguous after v_1 is removed. By contrast, if the next transfer were to move vertex $v_2 \in V$ to the hatched zone (Figure 6(b)), this transfer would satisfy the second condition of Theorem 1 but violate the first, because the unhatched zone no longer surrounds the hatched zone. To see how the second condition can be violated, consider the example depicted in Figure 6(c), where v_1 is to be transferred to the hatched zone. The first condition of Theorem 1 is satisfied because the hatched zone is surrounded by the unhatched zone. However, $R(v_1) \cap V(M_0(G) - \Pi_1(z(v_1)))$ contains the three unhatched neighbors of v_1, as well as v_0. The subgraph induced by these vertices has two components: one containing v_0 and the two horizontal neighbors of v_1, and a second containing only the unhatched neighbor positioned above v_1. Because the unhatched neighbors

Figure 6. Assessing contiguity constraints in a 6 6 grid-shaped region with two zones.



of v_1 are not all contained in one component, the second condition is violated, and Theorem 1 can be used to conclude that the unhatched district will become discontiguous if v_1 is removed.

Theorem 1 shows that assessing contiguity of zone $z\ v$ after the removal of v can be accomplished by examining only the vertices in $R\ v$. To ensure that transferring v to its new zone will neither eliminate zone $z\ v$ nor cause its new zone to become discontiguous, additional conditions must be added. Theorem 2 presents a complete set of conditions for identifying feasible local search transitions. Using geo-graphs, this vertex transfer is represented by creating two zoning functions: zone $z_1\ v$ contains v before the transition, and zone $z_2\ v$ contains v afterward.

THEOREM 2. *Let* $G_1\ V\ E\ B\ z_1$ *and* $G_2\ V\ E\ B\ z_2$ *be two geo-graphs describing the same region, where* $z_2\ x\ z_1\ x$ *for every* $x\ V\ v$, $z_2\ v\ k$ *for some* $k\ M\ G_1\ z_1\ v$, G_1 *has* $V\ j$ *for every* $j\ M\ G_1$, *and* G_1 *is zone-connected. Geo-graph* G_2 *is zone-connected with* $V\ j$ *for every* $j\ M\ G_2$ *if and only if the following conditions are satisfied*:

$N_{z_1\ v}\ v$ *is contained in a single component of the subgraph induced by* $R\ v\ V\ z_1\ v\ z_1\ v$ *in* G_1,

For every $1\ j\ z_1\ v$, $N_{z_1\ v}\ v$ *is contained in a single component of the subgraph induced by* $R\ v\ V\ M_0\ G_1\ _j\ z_1\ v$ *in* G_1,

$N_{z_1\ v}\ v$ *is not empty in* G_1,

$N_{z_2\ v}\ v$ *is not empty in* G_1.

The conditions established by Theorem 2 will not reject a feasible transition, nor will they allow an infeasible transition. The first two conditions of Theorem 2 are taken from Theorem 1 to establish that zone $z_1\ v$ remains contiguous. The third condition ensures that zone $z_1\ v$ is not eliminated by the transition (i.e., it still contains at least one vertex), while the fourth guarantees that zone $z_2\ v$ is contiguous. Both conditions can be verified by examining the vertices in $N\ v\ R\ v$, and therefore local search transitions can be classified as feasible or infeasible by examining only vertices whose unit boundaries share one point or more with the boundary of the unit being transferred.

While Theorem 1 considers the case when there might be surrounded zones, the conditions it checks become simpler when there are no surrounded zones. The following corollary establishes this simplification using the neighbor-connectedness of v in zone $z\ v$ (i.e., $N_{z\ v}\ v$ is contained in a single component of the subgraph induced by $R\ v\ V\ z\ v$).

COROLLARY 1. *Let* $G\ V\ E\ B\ z$ *be a zone-connected geo-graph with no surrounded zones. For any* $v\ V$, *the graph induced by* $V\ z\ v\ v$ *is connected if and only if* v *is neighbor-connected in* $z\ v\ M\ G$.

Like Theorem 1, this corollary determines whether it is possible to remove v from zone $z\ v$ without disconnecting it. Theorem 3 adds conditions to guarantee that v can be added to its new zone without disconnecting or eliminating any zones or creating any surrounded zones. This set of conditions can be used in geographic zoning applications that forbid the creation of holes.

THEOREM 3. *Let* $G_1\ V\ E\ B\ z_1$ *and* $G_2\ V\ E\ B\ z_2$ *be two geo-graphs describing the same region, where* $z_2\ x\ z_1\ x$ *for every* $x\ V\ v$, $z_2\ v\ k$ *for some* $k\ M\ G_1\ z_1\ v$, G_1 *has* $V\ j$ *for every* $j\ M\ G_1$, *and* G_1 *is zone-connected with no surrounded zones. Geo-graph* G_2 *is zone-connected with no surrounded zones and* $V\ j$ *for every* $j\ M\ G_2$ *if and only if the following conditions are satisfied*:

v *is neighbor-connected in zone* $z_1\ v$ *in* G_1,

$N_{z_1\ v}\ v$ *is not empty in* G_1,

$N_{z_2\ v}\ v$ *is not empty in* G_1,

The subgraph induced by $V\ z_2\ v$ *in* $H\ G_2$ *is connected.*

Only the fourth condition differs in purpose from the conditions in Theorem 2; this condition guarantees that no surrounded zones will be created by moving v to zone $z_2\ v$. Although Theorem 3 identifies feasible local search transitions when surrounded zones are forbidden, the ability to make such a transition depends on the existence of a current feasible solution. While zone-connected geo-graphs

are fairly simple to generate computationally, creating geo-graphs that lack surrounded zones is more difficult. Initial feasible solutions could be constructed by hand, but translating these hand-made solutions into a format usable in local search could be tedious, particularly when the number of units is large. An automated method avoids this obstacle. The final theorem in this paper provides a method for constructing zone-connected geo-graphs without surrounded zones. In particular, it shows how a new zone can be added to a geo-graph without being surrounded.

THEOREM 4. *Let $G_1 = (V, E, B, z_1)$ be a zone-connected geo-graph with no surrounded zones and $m(G_1) = k$, and let $v \in V$ be any vertex such that*:

- *v is neighbor-connected in $z_1(v)$, and $N_{z_1(v)}(v)$ is not empty,*
- *either there are two vertices, $x_1, x_2 \in R(v)$, such that $z_1(x_1) \neq z_1(x_2)$, or $B(v) \cap B(v_0) \neq \varnothing$.*

Define the function z_2, such that for all $x \in V - v$, $z_2(x) = z_1(x)$, and $z_2(v) = m(G_1) + 1$. Then the graph $G_2 = (V, E, B, z_2)$ is a zone-connected geo-graph with no surrounded zones, with $m(G_2) = m(G_1) + 1$.

Clearly, the only way to create a one-zone geo-graph is to place all units in the same zone. Using Theorem 4, new zones can be iteratively added to this single-zone solution until the desired number of zones is achieved. While the geo-graphs created by this mechanism will be very simple (i.e., $m(G) - 1$ zones with one unit each, and one zone with $|V| - m(G) + 1$ units), this starting point can be appropriately randomized by executing a number of randomly selected feasible transitions to reach a more reasonable initial state for local search.

Theorems 2 and 3 demonstrate how local search transitions that transfer $v \in V$ to a new zone can be classified as feasible or infeasible by examining only the vertices in $R(v)$. These examinations can be performed using "off the shelf" data structures and search algorithms. If an adjacency list realization of the planar subgraph induced by $R(v)$ for each $v \in V$ is stored, along with an adjacency matrix representation of the auxiliary graph $H(G)$, assessing the conditions of Theorem 2 for vertex v requires $O(m(G)^2 + m(G)|R(v)|)$ time; enumerating the pockets of zone $z_1(v)$ requires $O(m(G)^2)$ time using search, the first two conditions of Theorem 2 require $O(m(G)|R(v)|)$ time to search up to $m(G)$ subgraphs of $R(v)$ corresponding to these pockets, and the last two conditions require $O(|N(v)|)$ time. Similarly, assessing the conditions of Theorem 3 for vertex v requires $O(m(G)^2 + |R(v)|)$ time; the first three conditions can be assessed in $O(|R(v)|)$ time, while the last condition requires $O(|R(v)|)$ time to construct $H(G_2)$ from $H(G_1)$ and $O(m(G)^2)$ time to test the contiguity of $V' - z_2(v)$ in $H(G_2)$. Finally, the conditions of Theorem 4 for $v \in V$ can be assessed in $O(|R(v)|)$ time. While more efficient algorithms and data structures could improve these times, these simple implementations show that the time required to assess these conditions for vertex $v \in V$ is influenced primarily by the number of zones and the size of $R(v)$, neither of which will necessarily increase as the number of vertices increases.

7. Numerical Example

The benefits conferred by the geo-graph are entirely in the form of increased computational efficiency; using a geo-graph will not change the path that local search takes through the solution space, but will reduce the amount of computation required for local search to traverse this path. While the previous sections have discussed the theoretical contributions of the geo-graph model, its contributions when creating geographic zones depend on the characteristics of the zoned region. This section considers the creation of United States Congressional Districts in the state of Kansas following the 2000 Census, which involves dividing this state and its 2,688,418 residents into four districts. Basic units are available at several granularities, including counties and census blocks. Kansas contains 105 counties and 173,107 census blocks; while census blocks clearly provide a much larger solution space from which districts can be chosen, exploring this solution space requires significantly more computation than exploring the solution space afforded by counties. Geographic adjacency data were obtained from United States Census Bureau (2010b), while populations from the 2000 Census and district assignments for the 109th United States Congress were extracted from Missouri Census Data Center (2010).

Regardless of whether the basic units of G are chosen to be counties or census blocks, the number of zones remains constant; four United States Congressional Districts will be constructed from these units. Of particular interest, then, is how shifting from counties to census blocks impacts the size of $R(v)$. Each county has 3 to 8 units in $R(v)$, with a mean of 5.73 units, variance of 1.13 units^2, and median of 6 units. Each census block has 2 to 44 units in $R(v)$, with a mean of 7.09 units, variance of 6.60 units^2, and median of 7 units. Although moving from counties to census blocks represents more than a thousand-fold increase in the number of units, the average size of $R(v)$ increases by only 24%. The distribution of $|R(v)|$ for census blocks exhibits a much longer tail than the distribution for counties. However, relatively few units exhibit large values of $|R(v)|$; 80% of census blocks have $|R(v)| \leqslant 8$, 90% have $|R(v)| \leqslant 10$, and 99% have $|R(v)| \leqslant 16$. While some additional computation will be required to assess contiguity for census blocks, this additional computation is very small when compared to the growth in the number of basic units.

Of the 173,107 census blocks in Kansas, 1,764 have boundaries that are not a single simple closed curve. Of these blocks, 1,704 have holes. Such boundary violations can be addressed by merging these blocks with the blocks that reside in their holes. This merging reduces the size of the solution space without eliminating any feasible solutions; if the blocks in such a hole cannot form a district

on their own, then they must be in the same district as the block that surrounds them. In Kansas, the maximum population contained in any single hole is 738 people, about 0.1% of the population in the least populous district in Kansas, and certainly less than even the most lenient lower bound on district population. The 60 remaining units have multiple pieces; these violations can be addressed in several ways with only superficial impact on the solution space. After preprocessing the census blocks to eliminate these violations, 170,445 blocks remain, with the other blocks eliminated through merging. The statistics reported in the previous paragraph reflect the set of preprocessed census blocks.

Both simple search and geo-graph search evaluate district contiguity and hence will return the same outcome for any vertex $v \in V$ (i.e., whether zone $z(v)$ remains contiguous after removing unit v). Therefore, the only difference between these algorithms is how much computation they require to reach this outcome. While the size of $R(v)$ influences the time complexity of contiguity assessments for geo-graph search, the actual amount of computation required to carry out these assessments provides a more meaningful comparison with a simple search approach that runs in $O(|V(i)|)$ time when local search removes a block from zone $i \in M(G)$. Because both methods assess contiguity by searching some subgraph of G, the number of edges visited during these searches is a good measure of computation. For simple search, it is assumed that the subgraph induced by each $V(i)$ is stored separately and hence, simple search visits only edges with both endpoints in $V(i)$. For geo-graph search, it is assumed that the subgraph induced by each $R(v)$ is stored and hence, geo-graph search visits only edges with both endpoints in $R(v)$. However, any individual geo-graph search can visit only a subset of the vertices in $R(v)$ (i.e., those not contained in a particular set of zones); because vertices outside of this subset are also contained in $R(v)$, geo-graph search may visit edges with (1) both endpoints in the desired subset, or (2) only one endpoint in the subset and the other endpoint elsewhere in $R(v)$. Each search terminates successfully (i.e., $z(v)$ remains contiguous) when it visits all the vertices in $N_{z(v)}(v)$, and unsuccessfully (i.e., $z(v)$ becomes discontiguous) if it cannot visit all these vertices; for geo-graph search, this termination is due to Theorem 1, while for simple search this termination is proposed by Ricca and Simeone (2008). While Theorem 1 requires an additional search for each pocket of $z(v)$, the current districts in Kansas have no pockets, and therefore only a single geo-graph search is required for each block.

Under the district assignments of the 109th United States Congress, both search methods are applied to each census block in Kansas to determine whether removing that block violates district contiguity. A block $v \in V$ is omitted if (1) all of its neighbors are in district $z(v)$ (i.e., $N_{z(v)}(v) = N(v)$), or (2) the block has at most one neighbor in district $z(v)$ (i.e., $|N_{z(v)}(v)| \leqslant 1$). In the former case, the block cannot be moved into a new district; such a transition would cause this new district to become discontiguous. In the latter case, v satisfies Lemma 1 trivially; one can conclude that v can be removed from zone $z(v)$ without conducting a search. Under these restrictions, 1,940 of the 170,445 census blocks in Kansas require search. This proportion is relatively small, as the existing districts are noticeably compact, with relatively few blocks on the district boundaries. This high level of compactness is also noticeable in the outcomes of these searches, as 1,809 of the 1,940 census blocks tested can be removed without violating district contiguity.

Executing a geo-graph search on all 1,940 blocks required approximately 0.00289 seconds of CPU time on a 2.67-GHz quad-core processor running Windows 7 (computations were executed using a single core, as the implementation was not parallelized). By contrast, simple search required 2.96 seconds using breadth-first search (BFS) and 22.3 seconds using depth-first search (DFS); times are averaged over ten repetitions for BFS and DFS, and over 1,000 repetitions for geo-graph search. Table 1 summarizes the number of edges visited by each type of search. Unlike the statistics in Table 1, search times for simple search do include time to visit edges with only one endpoint in the subgraph, but these additional edges compose less than 1% of the edges visited in these searches and hence have minimal impact on computation time. Both BFS and DFS are implemented using a list of vertices to be explored; for BFS this list is a queue, while in DFS it is a stack. When a vertex is explored, the search visits each neighbor and adds it (if applicable) to the appropriate end of the list of vertices to be explored and hence an edge is considered visited when either of its endpoints is explored. From Table 1, it is clear that the geo-graph search visits a much smaller number of edges: 7.92 edges on average, as compared to 11,449 edges (BFS) or 91,894 edges (DFS) for simple search. Furthermore, the number of edges visited by geo-graph search is better controlled than simple search; each geo-graph search visits between 1 and 73 edges, while simple search may visit up to 396,660 edges. This discrepancy can be explained by the size of the subgraph being searched. Each geo-graph search considers a subgraph of at most 44 blocks (the maximum value of $|R(v)|$), while

Table 1. Statistics for the number of edges visited by geo-graph search, simple breadth-first search, and simple depth-first search (all statistics but sample size measure numbers of edges).

Search type	Geo-graph	Simple (BFS)	Simple (DFS)
Mean	7.92	11,449	91,894
Std. dev.	5.98	57,789	129,562
Min.	1	0	0
Max.	73	396,660	396,660
Median	7	37.5	5,105.5
Sample size	1,940	1,940	1,940

Table 2. Statistics for the number of additional edges visited by simple breadth-first and depth-first search as compared to geo-graph search under different conditions (all statistics but sample size measure numbers of edges).

	All blocks		Simple visits fewer		Cannot remove block		Can remove block	
	BFS	DFS	BFS	DFS	BFS	DFS	BFS	DFS
Mean	11,441	91,886	−1.23	−1.28	168,428	168,428	72.9	86,343
Std. Dev.	57,788	129,560	0.64	0.75	151,739	151,739	171.3	126,012
Min.	−6	−6	−6	−6	−6	−6	−2	−2
Max.	396,658	396,658	−1	−1	396,658	396,658	3,877	396,648
Median	30	5,099	−1	−1	198,860	198,860	28	3,672
Sample size	1,940	1,940	94	60	131	131	1,809	1,809

each simple search considers a subgraph with thousands of vertices.

For vertex $v \in V$, each search continues until it visits the set of vertices in $N_{z(v)}(v)$; in simple search, this set makes a very small portion of the blocks in a district, and hence, the number of edges visited is highly dependent on the path that the search takes though the district. This observation also explains why DFS visits, on average, many more edges than BFS: DFS tends to travel away from v and, by extension, away from the blocks in $N_{z(v)}(v)$. Of the 1,940 searches, DFS visits fewer edges than BFS in only 263. For both types of simple search, the median number of edges visited is much smaller than the mean, implying a skewed distribution where many searches visit relatively few edges and some searches visit a very large number of edges. Table 2 directly compares the number of edges visited in each type of search, showing that some of this skewness is explained by the outcome of each search; geo-graph search visits far fewer edges that simple search when removing the block violates contiguity, with more modest savings when contiguity is maintained. This table also shows that simple search visited fewer edges than geo-graph search for less than 5% of the investigated vertices, yielding only marginal improvement of 1.23 to 1.28 edges on average. Taken together, these results demonstrate the ability of geo-graph seach to visit relatively few edges when assessing district contiguity when compared to simple search.

8. Conclusion

The geo-graph model introduced in this paper provides an efficient structure for large graph partitioning problems when each vertex corresponds to a particular area of the plane, such as those encountered in geographic zoning problems. The geo-graph model supplies scale-invariant procedures to (1) evaluate the contiguity of the each partite set during local search and (2) numerically identify any holes that appear in each partite set. The scale invariance of contiguity assessments is tightly tied to the size of the augmented neighborhood, $R(v)$; the key result is that contiguity can be assessed by examining *only* these vertices. The size of $R(v)$ does not necessarily increase with the size of the graph; from a practical perspective, §7 shows that moving from counties to census blocks in the state of Kansas increases the number of units by a factor of more than 1,600 (from 105 counties to 170,445 census blocks), while the average size of $|R(v)|$ increases by a factor of only 0.24 (from 5.73 units to 7.09 units), demonstrating that the units considered in practical problems might scale well even when they are not strictly scale invariant. In contrast, the time complexity of assessing contiguity with simple search grows linearly with the size of the graph, discouraging practitioners from considering large districting problems that occur in practice.

The geo-graph model has been tailored to integrate duality-related correspondences between the plane graph summarizing unit adjacency and the plane graph describing unit boundaries. While this tailoring restricts this model from being applied to more general graphs, the computational savings provided by this tailoring can be substantial when the model can be applied. Furthermore, this structure remains modular to other aspects of partitioning. It does not restrict the user to a particular type of optimization objective, nor does it assume that any constraints other than contiguity will be imposed; although it enumerates the holes in each zone, these holes need not play a role in the actual optimization process. Modularity in objectives is particularly important in political districting, because different stakeholders in the districting process might want to consider a wide variety of different and conflicting objectives.

The time consumed to assess contiguity constraints makes up a portion of the total time spent in local search. As the number of units increases, using geo-graphs to evaluate these constraints will reduce computation when compared with simple search. From a broader perspective, increasing the number of units also leads to larger solution neighborhoods and possibly more local search iterations to algorithm termination. Geo-graphs cannot affect these facets of local search. Although geo-graphs can eliminate unnecessary computation associated with increasing problem size by exploiting the geometry of the problem, the growing computational costs required by other facets of the problem might be unavoidable, and hence choosing an appropriate granularity for the problem at hand remains a critical skill for successfully finding solutions of high quality.

Electronic Companion

An electronic companion to this paper is available as part of the online version at http://dx.doi.org/10.1287/opre.1120.1083.

Acknowledgments

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Air Force, the National Science Foundation, or the United States Government. The computational work was conducted with support from the Simulation and Optimization Laboratory at the University of Illinois. The authors thank Area Editor Alexander Shapiro, the associate editor, and the two anonymous referees for their insightful comments, which resulted in an improved manuscript. This research was supported in part by the National Science Foundation [IIS-0827540]. This material is based upon work supported in part while the second author served at the National Science Foundation. The second author was also supported in part by the Air Force Office of Scientific Research [FA9550-10-1-0387] .

References

Becker RI, Lari I, Lucertini M, Simeone B (1998) Max-min partitioning of grid graphs into connected components. *Networks* 32(2):115–125.

Bozkaya B, Erkut E, Laporte G (2003) A tabu search heuristic and adaptive memory procedure for political districting. *Eur. J. Oper. Res.* 144(1):12–26.

Bozkaya B, Erkut E, Haight D, Laporte G (2011) Designing new electoral districts for the city of Edmonton. *Interfaces* 41(6):534–547.

Butler D, Cain BE (1992) *Congressional Redistricting: Comparative and Theoretical Perspectives* (Macmillan Publishing Company, New York).

D'Amico SJ, Wang SJ, Batta R, Rump CM (2002) A simulated annealing approach to police district design. *Comput. Oper. Res.* 29:667–684.

di Cortona PG, Manzi C, Pennisi A, Ricca F, Simeone B (1999) *Evaluation and Optimization of Electoral Systems* (Society for Industrial and Applied Mathematics, Philadelphia).

Drexl A, Haase K (1999) Fast approximation methods for sales force deployment. *Management Sci.* 45(10):1307–1323.

Eppstein D, Italiano GF, Tamassia R, Tarjan RE, Westbrook J, Yung M (1992) Maintenance of a minimum spanning forest in a dynamic plane graph. *J. Algorithms* 13(1):33–54.

Fiduccia CM, Mattheyses RM (1982) A linear-time heuristic for improving network partitions. *DAC '82*: *Proc. 19th Design Automation Conf.* (IEEE Press, Piscataway, NJ), 175–181.

Frigioni D, Italiano GF (2000) Dynamically switching vertices in planar graphs. *Algorithmica* 28(1):76–103.

Garey MR, Johnson DS, Stockmeyer L (1976) Some simplified *NP*-complete graph problems. *Theoret. Comput. Sci.* 1(3):237–267.

Hansen P, Jaumard B, Meyer C, Simeone B, Doring V (2003) Maximum split clustering under connectivity constraints. *J. Classification* 20(2):143–180.

Kernighan BW, Lin S (1970) An efficient heuristic procedure for partitioning graphs. *The Bell System Tech. J.* 49(1):291–307.

Macmillan W (2001) Redistricting in a GIS environment: An optimisation algorithm using switching-points. *J. Geographical Systems* 3(2):167–180.

Minnesota Population Center (2004) National historical geographic information system: Pre-release version 0.1. Accessed May 25, 2011, http://www.nhgis.org/.

Missouri Census Data Center (2010) MABLE/Geocorr2K: Geographic correspondence engine with Census 2000 geography. Accessed October 18, 2011, http://mcdc.missouri.edu/websas/geocorr2k.html.

Nygreen B (1988) European assembly constituencies for Wales—Comparing of methods for solving a political districting problem. *Math. Programming* 42(1):159–169.

Ricca F (2004) A multicriteria districting heuristic for the aggregation of zones and its use in computing origin-destination matrices. *INFOR* 42(1):61–77.

Ricca F, Simeone B (2008) Local search algorithms for political districting. *Eur. J. Oper. Res.* 189:1409–1426.

Shi J, Malik J (2000) Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Machine Intelligence* 22(8):888–905.

Shirabe T (2005) A model of contiguity for spatial unit allocation. *Geographical Anal.* 37(1):2–16.

Shirabe T (2009) Districting modeling with exact contiguity constraints. *Environment and Planning B: Planning and Design* 36(6):1053–1066.

Tavares-Pereira F, Figueira JR, Mousseau V, Roy B (2007) Multiple criteria districting problems: The public transportation network pricing system of the Paris region. *Ann. Oper. Res.* 154:69–92.

United States Census Bureau (2000) Apportionment population and number of representatives, by state: Census 2000. Accessed May 27, 2011, http://www.census.gov/population/www/cen2000/maps/files/tab01.pdf.

United States Census Bureau (2010a) Census 2000: Summary file 1. Accessed May 25, 2011, http://www.census.gov/census2000/sumfile1.html.

United States Census Bureau (2010b) TIGER, TIGER/Line, and TIGER-related products. Accessed May 12, 2011, http://www.census.gov/geo/www/tiger/index.html.

United States Census Bureau (2011) Tallies of census blocks by state or state equivalent. Accessed May 12, 2011, http://www.census.gov/geo/www/2010census/census_block_tally.html.

Wang J-P (1998) Stochastic relaxation on partitions with connected components and its application to image segmentation. *IEEE Trans. Pattern Anal. Machine Intelligence* 20(6):619–636.

West DB (2001) *Introduction to Graph Theory*, 2nd ed. (Prentice Hall, Upper Saddle River, NJ).

Whitney H (1932) Non-separable and planar graphs. *Trans. Amer. Math. Soc.* 34(2):339–362.

Douglas M. King is a visiting lecturer in the Department of Industrial and Enterprise Systems Engineering at the University of Illinois. His current research interests include discrete optimization and mathematical modeling in applications related to public policy.

Sheldon H. Jacobson is a professor and director of the Simulation and Optimization Laboratory in the Department of Computer Science at the University of Illinois. He has a diverse set of basic and applied research interests, including problems related to optimal decision making under uncertainty, discrete optimization, causal inference with observational data, aviation security, public health policy (immunization, transportation and obesity, cell phone ban effectiveness), March Madness bracketology, and forecasting the outcome of the United States presidential election.

Edward C. Sewell is a Distinguished Research Professor of Mathematics and Statistics at Southern Illinois University at Edwardsville. His current research interests are combinatorial optimization and health applications.

Wendy K. Tam Cho is a professor in the Department of Political Science and Department of Statistics, and senior research scientist at the National Center for Supercomputing Applications, all at the University of Illinois at Urbana–Champaign.