Surrounds in Partitions

Matthew P. Dube
School of Computing and Information Science
University of Maine
5711 Boardman Hall, Orono, ME 04469-5711
matthew.dube@umit.maine.edu

Max J. Egenhofer
School of Computing and Information Science
University of Maine
5711 Boardman Hall, Orono, ME 04469-5711
max@spatial.maine.edu

ABSTRACT

Surrounds is a topological relation that can exist between two regions or between collections of regions in $\mathbb{R}^2$. This paper provides an algebraic construction for *surrounds* within a partition and provides a complementary graph-theoretic approach for the detection of the *surrounds* conditions created by the operations within the algebra. These two approaches are contrasted to one another. Constraints are placed upon *surrounds* to maintain certain algebraic benefits and the consequences of their relaxations are assessed.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS

General Terms

Management, Design, Human Factors, Theory.

Keywords

Discrete space, topology, spatial reasoning.

1. INTRODUCTION

The preposition *surrounds* refers to spatial configurations in which one object conceptually separates an embedding space into at least a pair of bounded and unbounded separations. The object that forms this separation *surrounds* anything that is located in the one of the bounded separation.

Reality and language expose us to wide varieties of *surrounds* as a term. The term is used in military strategy, where a discrete set of troops removes points of egress from an area. The term is used when objects nearly encircle other objects, such as how European, Middle Eastern, and African landmass *surround* the Mediterranean Sea. The term is also used for a landlocked country like Switzerland, an area itself that surrounds a German exclave Büsingen.

Mathematically, however, *surrounds* implies a more rigorously specified relation where the cells outside of a particular cell become disconnected into *n* separated collections of connected cells. While human languages operate in the realm of a conceptual *surrounds* preposition as previously commented, automated mapping systems and geographic information systems (GISs) do not operate in human linguistic realms. These types of systems need rigid decision standards to produce viable spatial queries, of which *surrounds* serves as an important one yet to be addressed.

This paper focuses on *surrounds* within the setting of partitions of the plane $\mathbb{R}^2$, that is, subdivisions of $\mathbb{R}^2$ into mutually exclusive cells. In partitions, two cells can be *disjoint* or they can *meet*, yet the other topological relations [14] known from simple regions—*overlap*, *coveredBy*, *inside*, *covers*, and *contains*—are impossible. The eighth region-region relation *equal* only holds between a cell and itself.

Political subdivisions are prime examples of partitions (taken at the granularity of smallest connected units) within which prototypical cases of *surrounds* relations are South Africa surrounding Lesotho, the mainland of Italy surrounding San Marino and the Vatican City, and the former German Democratic Republic surrounding West Berlin. In each case, the *host object* (the surrounding region) needs at least one *hole*, which in turn is filled by the surrounded object(s). While spatial relations related to containment have been studied extensively and have become a key ingredient of the formalized sets of topological relations [14,28], the spatial relations that rely on the concept of surrounding have received less attention [3,9,15,21,31]. Surrounding, however, differs from containment, as for containment, the two objects share a common interior, while for surrounding, the interiors are mutually exclusive (although the term *full physical containment* has been used as a superclass for both *surrounds* as well as *completely contains* [21].

While topological relations involving a holed region include specifications for surrounding configurations [14], the more involved cases of *surrounds* in multi-object spatial scenes, such as partitions, have not been studied.

(a) (b) (c)

Fig. 1: Three types of *surrounds* (differentiated by boundary contacts) in political subdivisions: (a) the Italian regions of Emilia-Romagna and Marche jointly surrounding the Republic of San Marino (b) thirteen Swiss Cantons surrounding the Canton Nidwalden, and (c) the Eurozone surrounding Liechtenstein.

Examples of such settings include the union of the Italian regions of Emilia-Romagna and Marche, jointly surrounding the Republic of San Marino (Fig. 1a), the union of the Swiss Cantons Jura, Neuchâtel, Vaud, Fribourg, Valais, Ticino, Graubünden, Glarus, St. Gallen, Zürich, Zug, Aargau, and Solothurn that surround the Canton Nidwalden (Fig. 1b), and the Eurozone (i.e., the countries that have adopted the Euro as their currency as of 2014) that surround Liechtenstein or a naïve conceptual model of Switzerland, assigning Büsingen (a German territory) from the European map to Switzerland (Fig. 1c).

The fundamental concepts of *surrounds* are identical, independent of whether there is a single object that *surrounds* something (Fig. 2a) or a set of objects that *surrounds* something else (Fig. 2b and 2c). The defining characteristic is that within the host region or set of host regions, a path exists that encircles the inner object. While *surrounds* may also apply to non-partitions (Fig. 2d), the detection of these paths is more involved and beyond the scope of this paper. Independent of whether the surrounding object is an individual or a collection, that object is labeled the *surround*.

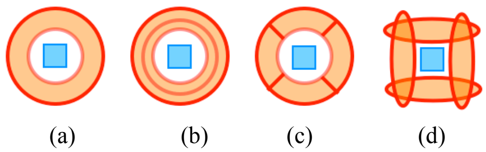


Fig. 2: Different scenarios of a surround given by (a) a single holed region, (b) a radial partition, (c) a sector partition, and (d) a scene with four overlapping regions.

The goal of this paper is to identify path-connected collections of cells that *surround* other path-connected collections of cells in a purely topological sense, and conversely, those collections of cells which are *surroundedBy* other cells. While we may describe other configurations with that terminology reflecting distance and position, the use of distance creates significant problems when size constraints are not immediately available with groups of objects (as opposed to a single object). The paper also differentiates *surrounds* based on boundary intersections between the *surround* and the *surrounded*.

The remainder of this paper continues with a review of the most closely related models for topological relations that may afford capturing *surrounds* in partitions (Section 2). Section 3 specifies *surrounds* relations, up to the development of an algebra for constructing complex scenes with *surrounds* relations. Section 4 discusses an assessment of particular constraints placed on the process. Section 5 provides an implementable graph-theoretic approach to detecting *surrounds*. Section 6 provides an algorithmic sketch of the *surrounds* detection mechanism. Section 7 provides conclusions and calls for future work in the field.

2. FORMALIZATIONS RELATED TO SURROUNDS

Partitions of space are a fundamental spatial abstraction, used as a model in many domains [18]. The formal foundation for spatial partitions [17] focuses on operations on entire partitions. A graph-theoretic model of spatial partitions discretizes these operations [26]. Three fundamental concepts relate to *surrounds* in partitions: (1) the concept of a partition of space as a setting that puts specific constraints on the objects involved as well as the spatial relations it features, (2) the structural setting that affords a *surrounds relation* by having holes in a surrounding region (empty or not), and (3) the spatial-relation models that address topological relations.

Without a hole—that is a separation of the exterior by some object—*surrounds* cannot occur. From an ontological perspective these kinds of holes are cavities [5]. Since the embedding space considered here is $\mathbb{R}^2$, not $\mathbb{R}^3$, other types of holes, such as hollows and tunnels, are not applicable. The semantics of physical containment relations focus on properties of the relations' domain and codomain [21], which imposes constraints on the possible occurrences of *surrounds*.

Considering essentially *fiat*, not *bona fide* objects [30] in partitions of $\mathbb{R}^2$, a complementary perspective is pursued here. Some accounts of *surrounds* definitions resort to inclusion in an object's convex hull [9]. While such semantics may be appropriate in specific ontologies (e.g., anatomy or other material physical settings [21]), it would lead to counter intuitive cases for partitions of fiat objects (e.g., it would specify that Switzerland *surrounds* Liechtenstein, ignoring that almost half of Liechtenstein's border is shared with Austria).

Since *surrounds* can be viewed as an encircling relation between objects in a space, that conception of *surrounds* carries no bearing of distance or shape inherent within it. In this restrictive form, it fits into the field of *topological spatial relations*, which remain invariant under topological transformations of the embedding space [8]. The two major models for binary topological relations—the *9-intersection* [15] (and its predecessor the 4-intersection [16]) and the *region connection calculus* (RCC) [28]—address *surrounds* relations differently. More extensive reviews of other related spatial-relation models can be found in survey articles [7,8,20] as well as a recent extension beyond binary topological relations [24]. RCC-8 allows for holed regions, but captures the resulting relations in holes like the relations outside a holed region. With the addition of more axioms, further distinctions are possible, however [4,6,19,34]. The 9-intersection, on the other hand, specifies the types of objects to which it applies and derives from the objects' topological properties what relations are distinguishable. In its most basic form, the 9-intersection focuses on *simple regions* (i.e., regularized, closed sets that are homeomorphic to 2-discs), disallowing such 2-dimensional objects as holed regions, pierced regions, or separations of regions (Fig. 3.).

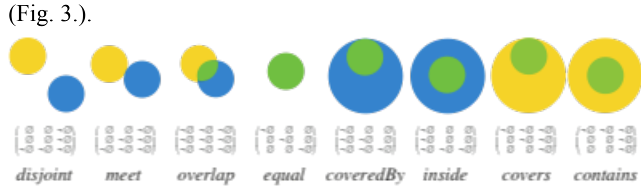


Fig. 3. RCC-8 and 9-intersection equivalent region-region relations.

Extending the relations' domain and codomain [25,29] increases the number of realizable empty/non-empty 9-intersection matrices, giving rise to capturing some *surrounds* relations [29], yet none of these approaches can distinguish a *disjoint* in a region's hole from a *disjoint* outside the region. Refinements of how non-empty intersections are recorded, including an incremental refinement for each separate intersection [23], go a step further, differentiating between the two types of *disjoint*, but since for a holed region the two exteriors are not explicitly identified as inner exterior and outer exterior, this approach falls short of assigning the actual *disjoint* relation. Such a distinction is only currently achieved with a compound-object model [10] that represents a single-holed region as the set difference of two regions, one *inside* the other, and records for each region the topological relation with respect to the target object [12,15].

With 9-intersection matrices [15] topological relations between entire partitions have been determined, without an expressive power to distinguish for two holed partitions whether one partition *surrounds* the other, or whether it is surrounded by the other partition. Bittner and Stell [5] developed a formal model of qualitative location, considering in a partition the topological relations between boundaries of cells.

Scene topology applies to relations between groups of objects, not only two. MapTree [32]—a graph-based model—and hull+o [24]—a model based on detailed topological relations [13] and relations with respect to regions' topological hulls [3]—allow for a complete, topologically correct reconstruction of scenes from their encoding. Both models have the power to capture *surrounds* relations, both between individual regions as well as in ensembles of regions. As such they come closest to providing a suitable foundation for specifying or implementing *surrounds* in partitions.

3. AN ALGEBRAIC APPROACH TO SURROUNDS

Before delving into *surrounds*, four terms are essential to creating the vocabulary used in this paper. These four terms are: *scene*, *topological hull*, *regularized collection*, and *cell*.

Definition 3.1: Let T be a topological space and $X_1 \ldots X_n$ be sets within T. T combined with $X_1 \ldots X_n$ is called a ***scene***.

Simply put, a scene is a combination of objects in a space to be analyzed. In the case of this paper, a scene consists of the embedding space $\mathbb{R}^2$ and a collection of regions that subdivide the space.

Definition 3.2: Let C be a connected region in space. C is called a ***cell***.

The theory of cells goes back to Alexandrov [2] and was furthered by Kovalevsky [22]. Cells partition space into bounded, connected regions.

Definition 3.3: Let A be a connected collection of cells. The ***topological hull* of *A***, denoted as $[A]$, is the smallest closed disc such that $[A] \supseteq A$ [24].

The topological hull is an operation that fills holes in regions, employing the philosophy of the compound object model [10]. While some cells (or collections of cells) might have holes in them, constructing or detecting holes is an important task. The topological hull helps to determine the presence of such a hole.

Definition 3.4: Let A be a collection of cells and let $[A]$ be its topological hull. If $A = [A]$, then A is called a ***regularized collection***.

Regularized collections are the basis for *surrounds* configurations from an algebraic perspective using the philosophy of the compound object model [10].

This paper builds on four distinct, elementary cases of *surrounds* relations involving two non-empty collections of partitioned cells (i.e., connected regions of space), A and B, with simply connected interiors. For B, it is further required that the union of all cells in B is equal to the topological hull of B [3,24], referred to here as a *regularized collection* (RC). Therefore, neither A nor B can be separated or touch in a finite number of points.

- A *surrounds* open, unoccupied space,
- A *surrounds* B, whose boundary lacks connection to A,
- A *surrounds* B, whose boundary has a partial connection to A, without separating the open space into two or more disconnected parts, and
- A *surrounds* only B, so that B's boundary is completely connected to A.

These four cases are called, respectively, *surroundsEmpty* (*sE*), *surroundsDisjoint* (*sD*), *surroundsMeet* (*sM*), and *surroundsAttach* (*sA*). Each of these four elementary cases has three components (Fig. 3):

- the topological *hull* [3,24] of host A, essentially filling any hole in A,
- the *hole* in the host A, and
- the *inner* collection of cells B (such that the exhaustive union of the collection's cell is equal to the collection's hull) that is *surroundedBy* A.

These three components are referred to as $s^*.hull$, $s^*.hole$, and $s^*.inner$, respectively, where $* \in \{E, D, M, A\}$ (Fig. 4). Only $sE.inner$ is empty; the other eleven components of any of the *surrounds* configurations yield a simple region. Each hull *contains* its hole, while each non-empty inner collection of cells is either *inside*, *coveredBy,* or *equal* to the hole. This configuration implies, by composition of "inner {*inside*, *coveredBy*, *equal*} hole" with "hole *inside* hull" [11] that each non-empty inner region is also *inside* the hull. A less-strict specification could allow holes to sit on the boundary of the object (i.e., hole {*inside*, *coveredBy*} hull). The implications of such a relaxation on the hole's position are discussed in Section 5.

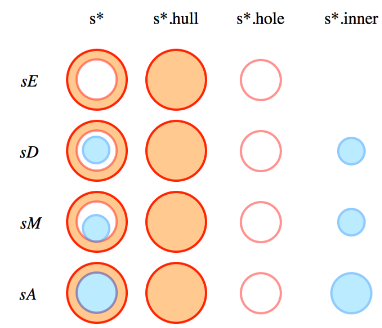


Fig. 4. The four cases of *surroundsEmpty*, *surroundsDisjoint*, *surroundsMeet*, and *surroundsAttach*, together with their hulls, holes, and inners.

3.1 Properties of Basic Surrounds Relations

The four *surrounds* relations have incompatible domains and codomains—while all four relations originate from a surround, none has a surround as its codomain as well. This property implies that none of these four relations is symmetric, reflexive, or transitive, and further, composition does not apply. The introduction of the converse relation to *r*, denoted by $\breve{r}$—that is, *surroundedByEmpty* ($\breve{sE}$), *surroundedByDisjoint* ($\breve{sD}$), *surroundedByMeet* ($\breve{sM}$), and, *surroundedByAttach* ($\breve{sA}$)—enables *compositions* over both collections.

;	sD	sM	sA
$\breve{sD}$	$d, m, o^*, eq, cB^*, i^*, cv^*, ct^*$	$d, m, o^* cB^*, i^*$	i^*
$\breve{sM}$	$d, m, o^* cv^*, ct^*$	$d, m, o^*, eq, cB^*, cv^*$	cB^*
$\breve{sA}$	ct^*	cv^*	eq

Fig. 4: The compositions of the converse relations with *surroundsEmpty* (*sE*), *surroundsDisjoint* (*sD*), *surroundsMeet* (*sM*), and *surroundsAttach* (*sA*), in a partition, yielding a disjunction of the relations *disjoint* (*d*), *meet* (*m*), *equal* (*eq*), and *overlap* (o^*), *coveredBy* (cB^*), *inside* (i^*), *covers* (cv^*), and *contains* (ct^*) if at least one of the two objects is a collection of more than one object.

Over the surround, the compositions $\breve{sI}\,; sJ \mid I, J \in \{D, M, A\}$ result either in simple region-region relations—*disjoint* (*d*), *meet* (*m*), or *equal* (*eq*), the latter only if the inferred relation is between the same partition cells—or also in the relations *coveredBy* (cB^*), *inside* (i^*), *covers* (cv^*), and *contains* (ct^*) if at least one object consists of more than a single cell, or *overlap* (o^*) if both objects have more than a single cell (Fig. 4). Only compositions involving sA and its converse $\breve{sA}$ have unique results. Compositions involving sE and $\breve{sE}$ are meaningless, since they do not involve a non-empty region.

3.2 Combinations of Surrounds Relations

Recursive applications of hull, hole, and inner yield more complex spatial partition scenes for *surrounds* relations. For the purpose of combining *surrounds* configurations, we introduce three orthogonal operations: (1) add hole $\odot$, (2) nest $\circledast$, and (3) add inner $\otimes$. Each of the three operations is specified by the intersection or union of two *surrounds* configurations, $X, Y \in \{sE, sD, sM, sA\}$, with constraints imposed on the existence of the inner regularized collections as well as the topological relations between the involved hulls, holes, and inner regularized collections (Eqs. 1-3).

$$X \odot Y = \overline{X \cap Y} \mid (X.hull\ equal\ Y.hull) \wedge (X.hole\ disjoint\ Y.hole) \quad (1)$$

$$X \circledast Y = \begin{cases} X \cup (X.inner \cap Y) & (X.inner \neq \emptyset) \wedge (X.inner\ equal\ Y.hull) \\ X & \text{otherwise} \end{cases} \quad (2)$$

$$X \otimes Y = \begin{cases} X \cup Y & (X.hull\ equal\ Y.hull) \wedge (X.hole\ equal\ Y.hole) \wedge (X.inner\ disjoint\ Y.inner) \\ X \cup Y & (X.hull\ equal\ Y.hull) \wedge (X.hole\ equal\ Y.hole) \wedge ((X.inner \neq \emptyset) \vee (Y.inner \neq \emptyset)) \\ X & \text{otherwise} \end{cases} \quad (3)$$

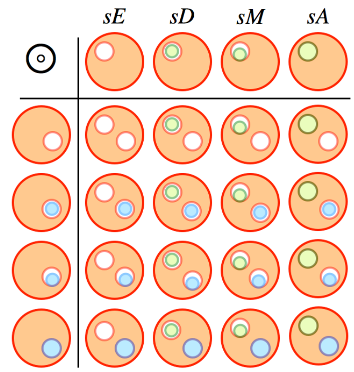
Figs. 5a-c provide visual examples for the semantics of these operations for each binary combination of sE, sD, sM, and sA.

As long as each hole, as well as each inner regularized collection, are considered a separate, distinct entity (i.e., they carry a different label), the sixteen combinations of adding a hole $\odot$ are unique, while nest $\circledast$ has identical results for all combinations that start with *surroundsEmpty*, whereas adding an inner regularized collection $\otimes$ has identical results for all combinations that start with *surroundsAttach*, as well as for $I \in \{D, M\}$, $sI \otimes sE = sI \otimes sA$. Uniqueness of combinations is further compromised if holes are interchangeable, or inner regularized collections are interchangeable among each other. In such a setting, for $I, J \in \{E, D, M, A\} \wedge I \neq J$ the combinations of $sI \odot sJ$ and $sJ \odot sI$ yield identical results.

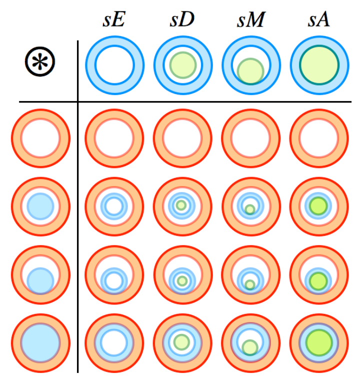
3.3 Algebra of *surrounds* Combinations

The algebraic properties—commutative, associative, distributive, and the identity element—for the three operators $\circledast$, $\odot$, and $\otimes$ show similarities, yet no two operators share exactly the same algebraic properties (Fig. 6).

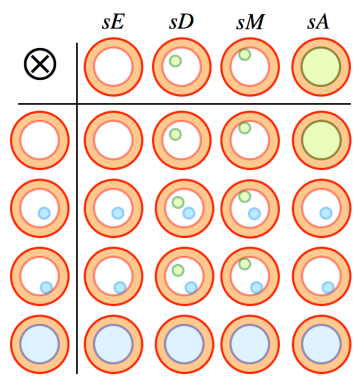
The idempotent operation *addHole* $\odot$ fixes the outer boundary between two constructions and places another hole within the original object. Removing any labeling considerations, for any $I, J \in \{E, D, M, A\}$, $s.I \odot s.J = s.J \odot s.I$, as the topological nature of the structure produces identical scenes; therefore, *addHole* is commutative. Also, for any $I, J, K \in \{E, D, M, A\}$, $s.I \odot (s.J \odot s.K) = (s.I \odot s.J) \odot s.K$, as all relations in question have a fixed outer boundary and all holes are enforced to be *disjoint*; therefore, *addHole* is associative. The *addHole* operator is, however, not distributive over the other two operators, because *nest* requires a fixed object to replace, while *addInner* requires a specified hole, both conditions that *addHole* cannot accommodate. Since *addHole* adds a hole to a pre-existing relation, its identity element is a regularly closed region (RC). Yet, this configuration is outside of the domain of *surrounds* relations.



(a)



(b)



(c)

Fig. 5: A visual display of the semantics of the operations (a) add hole, (b) nest, and (c) add inner into hole for the exhaustive binary combination of *surroundsEmpty*, *surroundsDisjoint*, *surroundsMeet*, and *surroundsAttach*.

	commutative	associative	distributive	identity element
addHole $\odot$	yes	yes	no	RC
nest $\circledast$	no	yes	no	RC
addInner $\otimes$	no	yes	no	*sE*

Fig. 6: Summary of the algebraic properties commutative, associative, distributive, as well as each operations' identity element, for the operations add hole $\odot$, nest region $\circledast$, and add inner region $\otimes$.

Since *nest* $\circledast$ produces a copy of another topological construction in place of the surrounded object in the first operand, it is inherently not commutative, because the surrounded boundary within the first operand becomes the outer boundary in the second operand. These two boundaries have a size mismatch; therefore, they cannot be exchanged to produce the same result. The operation *nest*, however, is associative in that it considers the inner most connected unit and replaces it, so that arbitrary groupings will not impact the result of the operation as long as the order is maintained. The mismatched dimensions of the operation automatically determine that *nest* is also not distributive over any operator from either side, leading to issues of which side of the other operators it is intended to apply to. Finally, *nest—like addHole*—has a regularized collection as its identity element.

The operation *addInner* $\otimes$ fixes the outer boundary between two constructions, but instead of generating a second hole within the original object, a copy of the internal structure of the second operand is placed within the first in a *disjoint* fashion. While *addInner* would appear to be commutative, it is not, because for constructions involving *surroundsAttach* it is impossible for a surrounded collection to be *disjoint* from the attached collection; therefore, $sA \otimes sD \neq sD \otimes sA$ and $sA \otimes sM \neq sM \otimes sA$. While $\otimes$ is no commutative, it is inherently associative. The operation is not distributive over either of the other operator, since *nest* causes confusion as to which of the internal relations undergoes the *nest* operation, and similarly *addInner* produces the conflict as to which of the holes gets the appended piece. Finally, sE is a left and right identity for *addInner*, because $\forall I \in \{E, D, M, A\}: sE \otimes sI = sI$ and $\forall J \in \{E, D, M, A\}: sJ \otimes sE = sJ$.

The final piece of the algebra over the three operators is to consider which of the operations take precedence over other operations, forming an implicit order of operations. Since *nest* is a replacement operation, it must occur first to avoid confusion with the particular inner collection in question. Since *addInner* operates upon the innermost hole available, it must occur before *addHole*, avoiding confusion as to which hole is the innermost. Finally, *addHole* is the last operation, appending additional internal structures. Proof of this sequence as the order of operations is beyond the scope of this paper. Fig. 7 displays a complex *surrounds* scene that can be uniquely constructed with these three operators in the sequence: $\{() \circledast \otimes \circledcirc\}$.

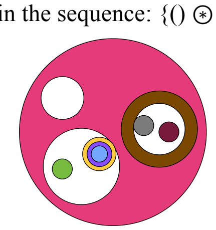


Fig. 7: A complex scene with multiple surroundings, created with the sequence $s.E \circledcirc (s.D \otimes (s.M \circledast s.A)) \circledcirc ((s.A \circledast s.M) \otimes s.D)$.

4. RELAXATION OF SURROUNDS CONSTRAINTS

In order to provide a focus on the most salient features of *surrounds*, the specifications (Section 3) intentionally restricted the flexibility of participating partitions in several specific properties (Fig. 8). The requirement of a surround to be simply connected, for instance, limits the configurations to have no separations, yet for such cases as political subdivisions separations are commonplace (Fig. 1b and 1c). This section examines the impact of relaxing these constraints in various ways.

C1: To account for separations in the specifications implies that the operation *addHole* needs an additional parameter that captures to which separation the added hole applies. Lifting the constraint, however, would also yield situations in which two or more separations may have a *0-meet* relation *in lieu* of being *disjoint*. Yet such *0-meet* configurations have the potential to create emerging holes, either through a multiple *0-meet* interactions by a single cell, or through a chain of cells connected by *0-meet*. C2 addresses the impact on the algebra of such an emerging hole.

C2: Allowing fringed holes in the specifications has no impact as long as the hole has only one boundary interaction with the host. With multiple interactions, however, the interior of the collection A is separated, so that the operation *addHole* needs an additional parameter that captures into which to part of the surround the hole is inserted.

	Constraint	Relaxed Constraint
C1	interior of collection simply connected	interior of collection potentially disconnected
C2	hole *inside* hull	hole *inside* v *0-coveredBy* hull
C3	hole *inside* hull	hole *inside* v *0*-coveredBy v *1-coveredBy* hull
C4	hole \ [inner] is connected	hull of inner disconnects inner void
C5	*disjoint* inners	*disjoint* v *0-meet* of inners
C6	*disjoint* inners	*disjoint* v *0-meet* v *1-meet* of inners
C7	*disjoint* holes	*disjoint* v *0-meet* of holes
C8	*disjoint* holes	*disjoint* v *0-meet* v *1-meet* of holes

Fig. 8. Constraints imposed on *surrounds* specifications (Section 3) and their potential relaxations.

C3: If the only hole of a collection were *1-coveredBy* the collection, then the hole and the host's exterior would be indistinguishable, and no *surrounds* configuration can be established. Therefore, C3 should not be relaxed.

C4: A disconnected inner void requires that the operation *addInner* needs an additional parameter that captures to which hole the added inner collection applies.

C5 and **C6:** Both result in scenarios that can split the hole into multiple connected components; therefore, creating another instance of C4. C6 also could produce *surroundsAttach*, if allowed.

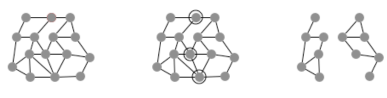
C7: Having a *0-meet* of holes is not an issue until those holes would disconnect the collection A, which is already addressed under C1 and C2.

C8: While two or more holes in a *1-meet* relation could be conceived (e.g., the political subdivision of the Lake of Constance among Germany, Switzerland, and Austria), it could also be treated in analogy to C3 as three non-distinguishable voids.

5. DETERMINING SURROUNDS IN PARTITIONS

The algebraic formalization of the different types of *surrounds* relations does not lead immediately to a framework that could be run over arbitrary partitions by contemporary GISs given that these systems are based on geometric, rather than symbolic, representations of spatial scenes [33]. In order to determine computationally whether, in a partition, a set of cells has one or more *surrounds* relations with respect to some other cells, corresponding definitions are needed on some implementable data structure. For this purpose, the partition is modeled as an adjacency graph of the partition's cells. Two fundamental graph concepts are used: (1) *path connectedness* [1] and (2) the *cutset*, usually defined for edges [1]. Here, however, a definition, centered on vertices, is the required conceptual model. A *vertex cutset* is any set of vertices whose removal would disconnect the graph, dissolving also any edges that start or end at the removed vertices (Fig. 8). This dissolved set of edges is a superset of at least one traditional cutset.

To create the full partition of a space, a systematic procedure is employed. Each path-connected collection of cells in the scene is unioned. With this basis set created, the first tool employed is the *topological hull* (Definition 3.3).



(a) (b) (c)

Fig. 9: Illustration of the concept of a vertex cutset: (a) a connected graph *G*, (b) a vertex cutset (highlighted vertices) of *G*, and (c) the disconnected graph resulting from removing the vertex cutset, together with the edges that start or end at the removed vertices.

With the computation of each *topological hull* of each connected collection, the next step is the definition of the holes within a collection of cells (Definition 5.1).

Definition 5.1: Let A be a collection of cells with topological hull $[A]$. If $[A]\backslash A = \emptyset$, A is an ***unholed cell***. If $[A]\backslash A = \neg\emptyset$, A is a ***holed cell***. The closure of each individual path-connected interior component of $\overline{[A]\backslash A}$ is a ***hole***. A is defined to have type ***interior*** while $\overline{[A]\backslash A}$ has type ***exterior***.

While a collection of cells might have a *hole* by Definition 5.1, that hole may not be devoid of type *interior* cells. The next aim is to produce a partition of space where no point is in more than one cell. Definitions 5.2 and 5.3 produce holes that are *separated exterior voids*, that is, they have no intersection with any type *interior* cell. Subsequently, scene *T* refers to a collection of cells of types *interior* or *exterior*.

Definition 5.2: Consider the collection of all cells and all holes within the scene *T*. Compute the set differences of all possible groupings. Any new sets created with the set difference are made available to the collection. The largest collection of these set differences that result in mutually exclusive cells are the ***bounded partitions*** of the scene *T*. Each cell of the *bounded partition* that does not have the relation *equal* with a cell of type *interior* is a ***separated exterior void*** and has type *exterior*.

The final cell to be identified in the scene *T* is the *outer void*, the unbounded component of the embedding space $\mathbb{R}^2$ (Definition 5.3).

Definition 5.3: The ***outer void*** of a scene *T*, denoted as V_T, is the set $\mathbb{R}^2\backslash B$, where B is the collection of *bounded partitions*. V_T is the portion of the exterior unbounded by any objects in the scene.

With the creation of the *outer void*, the entire space $\mathbb{R}^2$ is partitioned such that all non-boundary points are assigned to exactly one cell. From this point forward, all cells from a scene will be the cells of the *bounded partitions* and the *outer void*. The relation *strictly 1-meet* (Definition 5.4) leads to *adjacent* partitions (Definition 5.5).

Definition 5.4: Two collections of cells A and B have a ***1-meet*** relation if their o-notation [24] is $\overline{\partial}A: o_{\{B\}}(1, B, \emptyset)$ and $\overline{\partial}B: o_{\{A\}}(1, A, \emptyset)$ and all interactions *touch*.

Definition 5.5: Consider a topological scene *T* and its outer void V_T. The ***adjacency graph*** *G* is constructed by linking all cells (represented by a vertex) that have the relation *1-meet*. These cells are said to be ***adjacent***.

Definition 5.6: Let *G* be an adjacency graph and *V* a vertex cutset of *G*. All distinct connected parts of *G**V* are called **separated components** of *G**V*.

Definitions 5.1-5.4 enable the definition of a class of relations that map a vertex cutset *A* and a set of vertices *B* onto a set of vertices *C* that forms the class of *surrounds* relations: $r(G, A, B) \rightarrow C |\ A, B \subset G \wedge B \subset [A] \wedge B \cap A = \emptyset$. While this relation forms a viable *surrounds* class that is exhaustive of all strict *surrounds* relations, the relation can be subdivided based on the relationships within the graph between *A*, *B*, and the varying separated portions of the graph *G*.

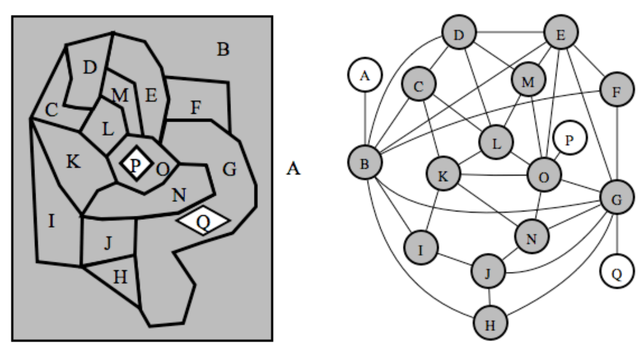
The simple algebra of *surrounds* combinations (Section 3.3) was limited to create operations that could be systematically and repeatedly applied, relying on four *surrounds* relations (*surroundsEmpty*, *surroundsAttach*, *surroundsDisjoint*, and *surroundsMeet*).

The following set of relations provides distinctions for seven different types of *surrounds* relations:

- *surroundsEmpty* (including that in Section 3)
- *surroundsAttach* (including that in Section 3)
- *surroundsAttachHole*
- *surroundsDisjoint* (including that in Section 3)
- *surroundsDisjointHole*
- *surroundsMeet* (including that in Section 3)
- *surroundsSplitPocket*

This set of these seven *surrounds* relations contains the *surrounds* relations of Section 3, but ultimately can achieve recursion and provide a mechanism for allowing multiple boundary contacts within a hole. This formalization can provide the vehicle for lifting many of the constraints within Section 4.

To display how the definitions in the section are applied, the same example partition of space is used (Fig. 10), comprising the general exterior A, fourteen type *interior* bounded partitions (B…O), and two type *exterior* bounded partitions (P and Q).



(a) (b)

Fig. 10. (a) Example partition and (b) corresponding adjacency graph to display of *surrounds definitions. Gray cells and vertices are of type interior, while white cells and vertices are of type exterior.**

Definition 5.7: Let G be an adjacency graph and A be a vertex cutset of G, and further let B be mutually exclusive from A and a subset of [A]. A ***surroundsEmpty*** B if and only if B is of type *exterior* and B is an entire separated component of $G \backslash A$.

surroundsEmpty (Definition 5.7; Fig. 11) is found when holes exist in a sensor network. The *surroundsEmpty* relation is useful as a detector for full coverage by a sensor network, or similarly for detecting that a sensor in a sensor network has ceased to function. Similarly, it is helpful for recognizing unaffected pockets by something like a pollution plume or an epidemic.

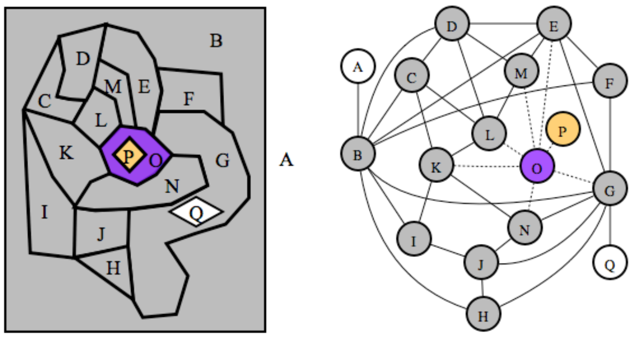


Fig. 11. *surroundsEmpty* displayed with the vertex cutset {O} (purple) and the surrounded cell P (cream).

Definition 5.8: Let G be an adjacency graph and A be a vertex cutset of G, and further let B be mutually exclusive from A and a subset of [A]. A ***surroundsAttach*** B if and only if all cells $C_i \subseteq B$ are of type *interior* and B is an entire separated component of $G \backslash A$.

surroundsAttach (Definition 5.8; Fig. 12.) is a very typical example in the political world and ultimately what we prototypically assume with the term *surrounds* in natural language: one object is completely up against another object with no way out. This happens politically with such countries as Lesotho, the Vatican City, and San Marino. It also happens between groups of countries toward another country (e.g., Switzerland as a landlocked country) and groups of countries to other countries (e.g., Switzerland and Liechtenstein being surrounded in this way by Eurozone countries).

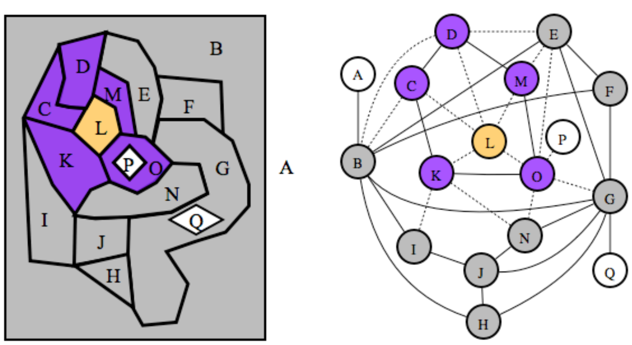


Fig. 12. *surroundsAttach* in the example partition. This instance would be recursive as there is still a separation of the graph ({A,B,E,F,G,H,I,J,N,Q} on one side; P on the other).

Definition 5.9: Let G be an adjacency graph and A be a vertex cutset of G, and further let B be mutually exclusive from A and a subset of [A]. A ***surroundsAttachHole*** B if and only if B is an entire separated component of $G \backslash A$ and at least one cell $C_i \subset B$ is of type *exterior*, and all such cells of type *exterior* are not adjacent to A within G, or B is a proper subset of a separated component such that no remaining vertices within the separated component are adjacent to A.

surroundsAttach (Definition 5.8; Fig. 12) and *surroundsAttachHole* (Definition 5.9; Fig. 13) differ in that, while the outer boundary of the separated component is covered, there remains an inner boundary. In some cases, that inner region may be another object of interest, while in other cases it may be of type *exterior*.

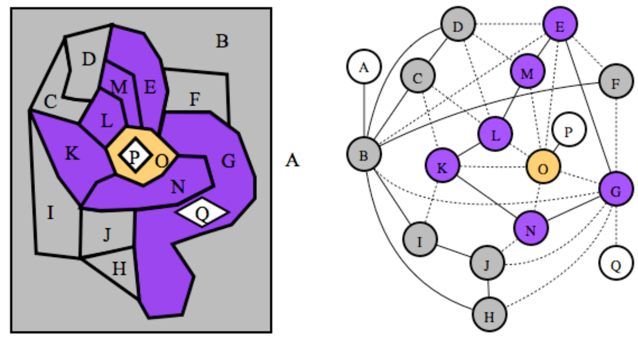


Fig. 13. *surroundsAttachHole*, in this case with an isolated component of type *exterior*.

Definition 5.10: Let G be an adjacency graph and A be a vertex cutset of G, and further let B be mutually exclusive from A and a subset of [A]. A ***surroundsDisjoint*** B if and only if B is a proper subset of a separated component of $G \backslash A$, all cells $C_i \subseteq B$ are of type *interior*, and no such cell has adjacency to A within G.

surroundsDisjoint (Definition 5.10; Fig. 14.) represents a layer around a cell. This layer serves as a buffer between the host collection and the surrounded collection. For instance, one could view Kansas, Missouri, Nebraska, and Iowa as the center of the United States, separated from any point of egress by at least two other states.

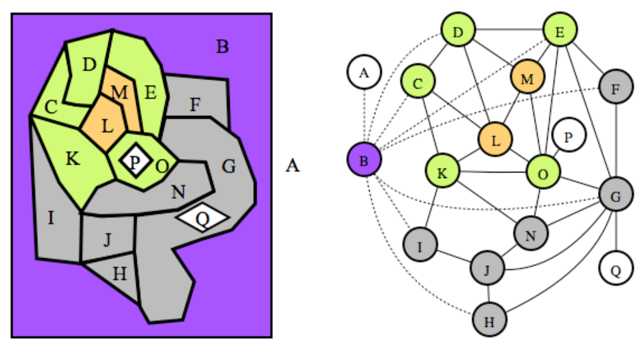


Fig. 14. *surroundsDisjoint* in the example partition. The green cells C, D, E, K, O represent the buffer between cells L and M (the surrounded cells) and B (the host cell). No path from L or M to B is free of this set of vertices.

Definition 5.11: Let G be an adjacency graph and A be a vertex cutset of G, and further let B be mutually exclusive from A and a subset of [A]. A ***surroundsDisjointHole*** B if and only if B is a proper subset of a separated component of $G \backslash A$, no cell $C_i \subseteq B$ has adjacency to A within G, and $G \backslash A \backslash B$ has more separated components than $G \backslash A$.

surroundsDisjoint (Definition 5.10; Fig. 14.) and *surroundsDisjointHole* (Definition 5.11; Fig. 15.) differ similarly to *surroundsAttach* and *surroundsAttachHole*. They present a guaranteed opportunity for recursion as cell *B* in this case serves as a buffer to the new separated component itself.

Definition 5.12: Let *G* be an adjacency graph and *A* be a vertex cutset of *G*, and further let *B* be mutually exclusive from *A* and a subset of [*A*]. *A* ***surroundsMeet*** *B* if and only if *B* is a proper subset of a separated component of *G**A*, at least one cell $C_i \subseteq B$ has adjacency to *A* within *G*, all cells in *B* are of type *interior, B* induces no more separated components of *G**A*, and at least one remaining vertex from the separated component is adjacent to *A*.

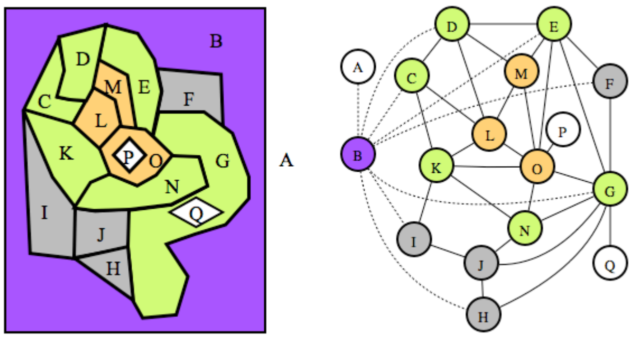


Fig. 15. *surroundsDisjointHole* in the example partition. In this case, the isolated component is of type *exterior*.

surroundsMeet (Definition 5.12; Fig. 16.) is similar to *surroundsDisjoint* (Definition 5.10; Fig. 14.) in that the separated component is not exhausted, but the fundamental difference between the two is that *surroundsMeet* requires boundary contact, whereas *surroundsDisjoint* excludes it. For instance, the world's oceans surround the Mediterranean Sea in this way.

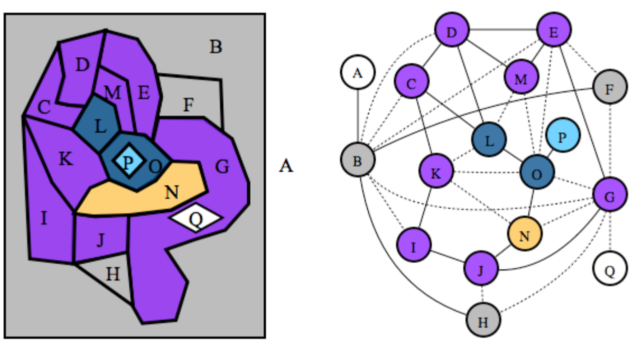


Fig. 16. *surroundsMeet* in the example partition. The blue vertices L, O, and P represent cells that vertex N can access without passing through the host collection. Some of these vertices (L and O) have adjacency to the host collection.

Definition 5.13: Let *G* be an adjacency graph and *A* be a vertex cutset of *G*, and further let *B* be mutually exclusive from *A* and a subset of [*A*]. *A* ***surroundsSplitPocket*** *B* if and only if *B* is a proper subset of a separated component of *G**A*, at least one cell $C_i \subseteq B$ has adjacency to *A* within *G*, all cells in *B* are of type *interior*, and *B* induces at least one additional separation of *G**A*.

surroundsSplitPocket (Definition 5.13; Fig. 17.) and *surroundsMeet* (Definition 5.12; Fig. 16.) differ in that *surroundsSplitPocket* creates at least one additional separated component in the resulting vertex cutset. Either of these relations guarantees the result to be recursive; however, *surroundsSplitPocket* guarantees multiple opportunities for recursion.

Given that the domain of the class of *surrounds* relations is all vertex cutsets and its codomain is all components separated by the vertex cutset, *surrounds* as a relation is recursive so long as additional separated components remain after the application of a *surrounds* relation. Once all holes in a collection of cells *A* are filled, the relation is at its final level.

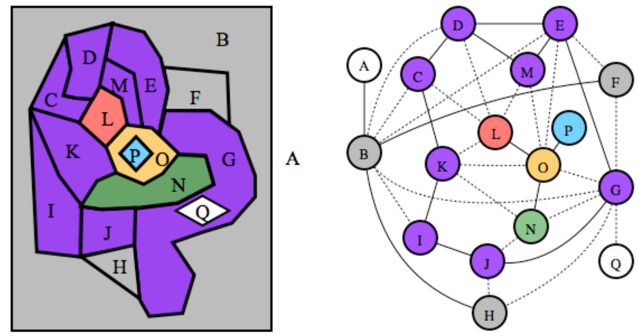


Fig. 17. *surroundsSplitPocket* in the example partition. In this case, the addition of O to the vertex cutset (C,D,E,G,I,J,K,M) would induce two extra separated components in the graph *G*.

Some relaxations impact the graph formalization as well. Whenever a relaxation would allow a *0-meet* or a *0-coveredBy*, the graph formalization loses its ability to detect a fringed surround as a *surrounds* relation through the vertex cutset. The recursive nature of the graph formalization and its construction based solely on a vertex cutset makes it more flexible in accounting for separations and multiple hole scenarios. What it allows for has the added drawback that it takes away the ability to carve out features. Definition 5.14 adds this carving ability.

Definition 5.14: Let *A* be a vertex cutset of a graph *G*. Consider the induced subgraph of *A* from *G*. If any set of this induced subgraph with no edge connection to a separated component of *G**A* in *G* is a vertex cutset of the induced subgraph, then this vertex cutset and the separated components contained in its topological hull can be removed to form *A**, imposing an additional separated component of *G**A*.

This additional definition satisfies that the refined version *A** is still a vertex cutset, still creates additional separated components, and further maintains the prior separated components from the regular version of *A*.

6. AN ALGORITHM FOR SURROUNDS

Using the rules from Sections 5, an algorithmic design for *surrounds* can be constructed for adjacency-aware spatial information systems such as ESRI's ArcGIS suite and OracleSpatial, amongst others.

For an algorithm to decide on whether or not a collection of cells *surrounds* another collection of cells, information about the structure of the space, the types of the potentially surrounded vertices, the number of separated components after removal of the host, and the adjacency of both the remaining separation and the surround to the host (Fig. 18).

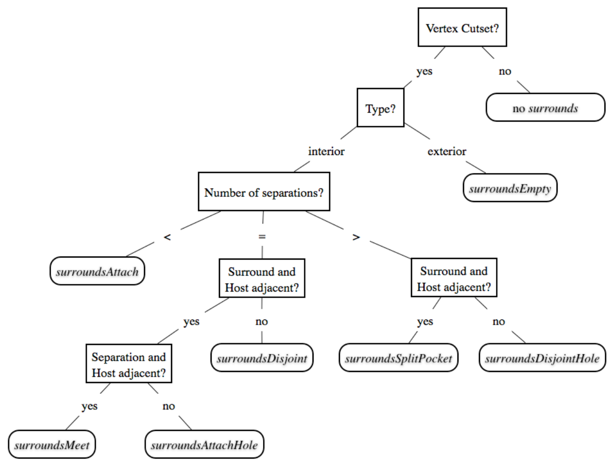


Fig. 18. *surrounds decision tree.**

Under this decision tree structure, *surrounds* can be codified as a queryable relation within spatial information systems. Algorithm 6.1 displays the pseudocode of such an algorithm.

Algorithm 6.1: Pseudocode for Surrounds Classification

Given: *adjacencyGraph* [*n* x *n* matrix], *vertexType* [*n* list]
Input: *host* [*m* list], *surrounded* [*k* list]
Compute: # of connected components of *adjacencyGraph* (*a*)
Remove *host* from *adjacencyGraph* → *hostGraph*
Compute: # of connected components of *hostGraph* (*b*)
If *b* > *a*:
  If *vertexType*[*surrounded*] = "exterior"
    Output "surroundsEmpty"
  Else:
    Select connected component containing *surrounded* → *comp*
    Remove *surrounded* from *hostGraph* → *surroundGraph*
    Compute: # of connected components of *surroundGraph* (*c*)
      If *c* < *b*:
        Output "surroundsAttach"
      If *c* = *b*:
        If *surrounded* adjacent to *host* in *adjacencyGraph*:
          Select connected component of *surroundGraph* within *comp* → *inner*
          If *inner* adjacent to *host*:
            Output "surroundsMeet"
          Else: Output "surroundsAttachHole"
        Else: Output "surroundsDisjoint"
      If *c* > *b*:
        If *surrounded* adjacent to *host* in *adjacencyGraph*:
          Output "surroundsSplitPocket"
        Else: Output "surroundsDisjointHole"
Else: Output "no"

7. CONCLUSIONS AND FUTURE WORK

This paper produced an algebraic form of *surrounds* in partitions based on the premises of two scene topology tools [24]: the topological hull and the *o*-notation. With four basic types of *surrounds* and three operations to combine such *surrounds* types, an algebra was created for constructing complex *surrounds* configurations. Furthermore, a graph version of *surrounds* was created based on an adjacency paradigm of *1-meet*, allowing for non-separating fringed holes to be considered and for *surrounds* to actively be computed within existing adjacency frameworks based solely on the concept of *vertex cutset*. This version of *surrounds* serves as a detection mechanism underneath the current standards of geometric storage within contemporary GISs.

While *surrounds*, as tackled in this paper, is very flexible, there are alternative versions of *surrounds* that can be explored. While this paper focused on cells in partitions (effectively representing regions or collections thereof), lines and points can also be surrounded (e.g., Four Corner is a point surrounded by the States of Arizona, Colorado, New Mexico, and Utah). Since these ideas do not integrate well with partitions of space, new definitions and paradigms would be required to accommodate an integrated framework covering holistically regions, lines, and points.

Another such *surrounds* scenario can be found in separated objects that together surround another object. For instance, consider two donuts, one inside the other's hole, and a third donut between the two (Fig. 19a). Since the algebra for constructing scenes with *surrounds* relations (Section 3) does not yet include objects made up of multiple separated collections of cells, this nested donut scenario is not yet constructable. In future work, the algebra needs to be extended to account for such self-nested scenarios. On the other hand, the detection mechanism of *surrounds* relations is generic enough to determine that such a combination of donuts *surrounds* the third donut, specifically *surroundsDisjoint* (Fig. 19b).

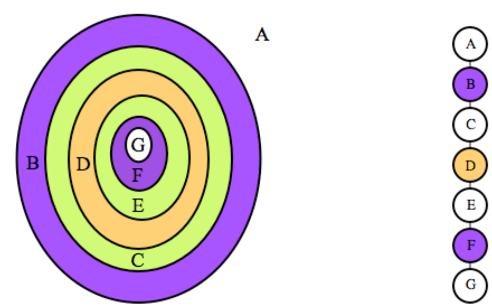


Fig. 19. The union of C and G surroundsdisjoint E, much like C surroundsDisjoint C.

Preparata and Shamos [27] proposed methods for geometric searching through files. In their work, they proposed different types of queries: *single-shot* and *repetitive-mode*. While *surrounds* can be a *single-shot query*, the stackability of *surrounds* forms a need for a *repetitive-mode query*. Integrating the *surrounds* query into a geometric search via these methods could be a fruitful endeavor in settings such as emergency planning with ArcMap using the Polygon Neighbors Analysis tool.

8. ACKNOWLEDGMENTS

Matthew Dube is an IGERT fellow (funded by NSF grant DGE-0504494, PI: Kate Beard) and supported by a UMaine Michael J. Eckardt Dissertation Fellowship. Max Egenhofer's work was partially supported by NSF Grant IIS-1016740. The anonymous comments of reviewers are also acknowledged, creating meaningful refinements to the paper.

9. REFERENCES

[1] Adams, C., Franzosa, R.: Introduction to Topology. Pearson Prentice Hall (2008)

[2] Alexandrov, P.: Combinatorial topology. Vol. 1. Courier Dover Publications (1956)

[3] Bennett, B., Chaudhri, V., Dinesh, N.: A Vocabulary of Topological and Containment Relations for a Practical Biological Ontology. In: Tenbrink, T., Stell, J., Galton, A., Wood, Z. (eds.) Spatial Information Theory COSIT '13, LNCS 8116: 419-37 (2013)

[4] Bittner, T., Stell, J.: A Boundary-Sensitive Approach to Qualitative Location. Annals of Mathematics and Artificial Intelligence 24(1-4): 93-114 (1998)

[5] Cassati R., Varzi, A.: Holes and Other Superficialities. MIT Press (1994)

[6] Cohn, A., Bennett, B., Gooday, J., Gotts, N.: Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. GeoInformatica 1(3), 275-316 (1997)

[7] Cohn, A., Hazarika, S.: Qualitative Spatial Representation and Reasoning: An Overview. Fundamenta Informaticae 46(1-2), 2-32 (2001)

[8] Cohn, A., Renz, J.: Qualitative Spatial Representation and Reasoning. In: van Hermelen, F., Lifschitz, V., Porter, B. (eds.) Handbook of Knowledge Representation, pp. 551-596 (2008)

[9] Donnelly, M.: Containment Relations in Anatomical Ontologies, In: Friedman, C., Ash, J., Tarczy-Hornoch, P. (eds.) AMIA 2005 Annual Symposium, pp. 206 -210 (2005)

[10] Egenhofer, M.: A Reference System for Topological Relations between Compound Spatial Objects. In: Heuser, C., Pernul, G. (eds.) Advances in Conceptual Modeling—Challenging Perspectives, ER 2009 Workshops, LNCS 5833, pp. 307-316 (2009)

[11] Egenhofer, M.: Deriving the Composition of Binary Topological Relations. Journal of Visual Languages and Computing 5(2), 133-149 (1994)

[12] Egenhofer, M., Clementini, E., Di Felice, P.: Topological Relations between Regions with Holes. International Journal of Geographical Information Systems 8(2), 129-142 (1994)

[13] Egenhofer, M., Franzosa, R.: On the equivalence of topological relations. International Journal of Geographical Information Systems 9(2): 133-152 (1995)

[14] Egenhofer, M., Franzosa, R.: Point-Set Topological Spatial Relations. International Journal of Geographical Information Systems 5(2), 161-174 (1991)

[15] Egenhofer, M., Herring, J.: Categorizing Binary Topological Relationships Between Regions, Lines, and Points in Geographic Databases, Department of Surveying Engineering, University of Maine, Orono, ME (1991)

[16] Egenhofer, M., Vasardani, M.: Spatial Reasoning with a Hole. In: Winter, S., Duckham, M., Kulik, L., Kuipers, B. (eds.) Spatial Information Theory, 8th International Conference, COSIT 2007, Melbourne, Australia, LNCS 4736, pp. 303-320 (2007)

[17] Erwig, M., Schneider, M.: Partition and Conquer. In: Hirtle, S., Frank, A. (eds.) Spatial Information Theory: A Theoretical Basis for GIS, International Conference COSIT '97, Laurel Highlands, PA, LNCS 1329, pp. 389-407 (1997)

[18] Frank, A.: Spatial Concepts, Geometric Data Models, and Geometric Data Structures. Computers & Geosciences 18 (4), 409-417 (1992)

[19] Galton, A.: Lines of sight. AI and Cognitive science 94: 103-113 (1994)

[20] Galton, A.: Spatial and Temporal Knowledge Representation. Earth Science Informatics 2(3), 169-187 (2009)

[21] Hahmann,T., Brodaric, B.: Kinds of Full Physical Containment. In: Tenbrink, T., Stell, J., Galton, A., Wood, Z. (eds.) Spatial Information Theory COSIT '13, Scarborough, UK, LNCS 8116: 397-417 (2013)

[22] Kovalevsky, V.: Geometry of Locally Finite Spaces. Publishing House, Berlin, Germany 12 (2008)

[23] Kurata, Y.: The 9+-intersection: A universal framework for modeling topological relations. Geographic Information Science. Springer Berlin Heidelberg: 181-198 (2008)

[24] Lewis, J., Dube, M., Egenhofer, M. The Topology of Spatial Scenes in R2. In: Tenbrink, T., Stell, J., Galton, A., Wood, Z. (eds.) Spatial Information Theory COSIT '13, Scarborough, UK, LNCS 8116: 495-515 (2013)

[25] Li, S.: A complete classification of topological relations using the 9-intersection method. International Journal of Geographical Information Science 20(6): 589-610 (2006)

[26] McKenney, M., Schneider, M.: Spatial Partition Graphs: A Graph Theoretic Model of Maps. In: Papadias, D., Zhang, D., Kollios, G. (eds.) Advances in Spatial and Temporal Databases, 10th International Symposium, SSTD 2007, Boston, MA, LNCS 4605, pp. 167-184 (2007)

[27] Preparata, F. Shamos, M.: Computational Geometry. Springer, New York (1985)

[28] Randell, D., Cui, Z., Cohn, A.: A Spatial Logic Based on Regions and Connection. In: 3rd International Conference on Knowledge Representation and Reasoning, pp. 165-176 (1992)

[29] Schneider, M., Behr, T.: Topological Relationships Between Complex Spatial Objects. ACM Transactions on Database Systems 31(1): 39-81 (2006)

[30] Smith, B.: Fiat Objects. Topoi 20(2): 131-148 (2001)

[31] Vasardani, M., Egenhofer, M.: Comparing Relations with a Multi-Holed Region. In: Hornsby, K., Claramunt, C., Denis, M., Ligozat, G. (eds.) Conference on Spatial Information Theory (COSIT'09), LNCS 5756, pp. 159-176 (2009)

[32] Worboys, M.: The Maptree: A Fine-Grained Formal Representation of Space. In: Xiao, N., Kwan, M.-P., Goodchild, M., Shekhar, S. (eds.) Geographic Information Science. LNCS 7478, 298-310 (2012)

[33] Worboys, M., Duckham, M.: GIS: a computing perspective. CRC press, 2004.

[34] Yang, J., Fu, Q., Liu, D.: A model for representing topological relations between simple concave regions. Computational Science–ICCS 2007. LNCS 4487, pp. 160-167 (2007)