*ECAI 2014*
*T. Schaub et al. (Eds.)*
*© 2014 The Authors and IOS Press.*
*This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.*
*doi:10.3233/978-1-61499-419-0-795*

# Declarative Spatial Reasoning with Boolean Combinations of Axis-Aligned Rectangular Polytopes

**Carl Schultz** and **Mehul Bhatt**[1]

**Abstract.** We present a formal framework and implementation for declarative spatial representation and reasoning about the topological relationships between boolean combinations of regions (i.e., *union, intersection, difference, xor*). Regions of space here correspond to arbitrary axis aligned n-polytope objects, with geometric parameters either fully grounded, partially grounded, or completely unspecified. The framework is implemented in the context of **CLP($\mathcal{QS}$)**[2], a constraint logic programming based declarative spatial reasoning system providing support for geometric and qualitative spatial abstraction and inference capabilities.

We demonstrate that our method can solve *packing*, *contact*, *containment*, and *constructive proof* problems that are unsolvable using standard relational algebraic approaches for qualitative spatial reasoning (QSR). Our approach is driven by general accessibility of spatial reasoning via KR languages for their application in domains such as *design, geography, robotics, and cognitive vision*.

## 1 Motivations

Knowledge representation and reasoning about *space* may be formally interpreted within diverse frameworks such as: (a) geometric reasoning & constructive (solid) geometry [Kapur and Mundy, 1988]; (b) relational algebraic semantics of 'qualitative spatial calculi' [Ligozat, 2011]; and (c) by axiomatically constructed formal systems of mereotopology and mereogeometry [Aiello et al., 2007]. Independent of formal semantics, commonsense spatio-linguistic abstractions offer a human-centred and cognitively adequate mechanism for logic-based automated reasoning about spatio-temporal information [Bhatt et al., 2013].

Formalizations (e.g., logical, relational-algebraic) of space and development of tools for efficiently reasoning with spatial information is a vibrant area of research within knowledge representation and reasoning (KR) [Renz and Nebel, 2007, Bhatt et al., 2011a,b]. Research in qualitative spatial representation and reasoning has primarily been driven by the use of constraint-based reasoning algorithms over an infinite (spatial) domain to solve *consistency problems* in the context of *qualitative spatial calculi* [Ligozat, 2011]. The key idea behind the development of qualitative spatial calculi has been to partition an infinite quantity space into finite disjoint categories, and utilize the special *relational algebraic properties* of such a partitioned space for reasoning purposes [Guesgen, 1989, Ligozat, 2011]. Similarly, logic-based axiomatisations of topological and mereotopological space and a study of their general computational characteristics have also been thoroughly investigated [Aiello et al., 2007, Borgo, 2009, Kontchakov et al., 2013]. In spite of significant developments in the theoretical foundations of spatial representation and reasoning, and the availability of relational-algebraically founded spatial reasoning algorithms [Condotta et al., 2006], what is still missing in the (geometric and qualitative) spatial reasoning community is a unifying KR-based framework of space, and an implementation of geometric and qualitative spatial representation and reasoning in a manner that would seamlessly / natively integrate with general KR languages and frameworks in artificial intelligence (AI). Addressing this gap, recent research initiatives have started to address formalisations of space / spatial reasoning within declarative KR languages such as logic programming and constraint logic programming [Bhatt et al., 2011b, Schultz and Bhatt, 2012]; the approach underlying this line of work, manifested by the CLP(QS) spatial reasoning system, marks a clear departure from other reasoning tools by its use of the constraint logic programming framework for formalising the semantics of geometric and qualitative spatial representation and reasoning. The approach has been driven by and found applicability in a range of AI systems concerning visuo-spatial language, learning, and cognition [Bhatt et al., 2013] (e.g., a prime recent example here being *architectural design cognition* [Bhatt et al., 2014]).

This paper is situated in the context of the CLP($\mathcal{QS}$) declarative spatial reasoning framework – currently, CLP($\mathcal{QS}$) provides topological reasoning capabilities over uniform-dimensional 2D regions that can be interpreted as convex polygons, and additionally, supports orientation reasoning with intrinsically-oriented point objects[3]. We extend CLP($\mathcal{QS}$) with the capability to reason about boolean combinations of regions —i.e., *union, intersection, difference, xor*— natively within the constraint logic programming framework. Regions of space within our framework correspond to arbitrary n-polytope objects, with geometric parameters either *fully grounded*, *partially grounded*, or even *completely unspecified*. In other words, our approach supports mixed qualitative-quantitative spatial reasoning, which is desirable since a purely geometric approach is not satisfactory when only incomplete information is available about the regions, or when the information about relationships between regions is *qualitative* (e.g., in domains involving space and natural language). An additional feature of our approach is that it is possible to freely mix different *types* of spatial relations (e.g., *topology, orientation, distance, shape*) and different *types* of objects (e.g., *rectangles, cuboids, spheres, points*). We show decidability of our algorithm for solving the consistency problem on spatial constraint networks with a finite number of objects. Furthermore, we demonstrate the implementation by its application for solving classes of *packing* [Simonis and O'Sullivan, 2008], *contact*, *containment*[4], and *constructive proof* problems (Section 4) that are unsolvable using standard relational algebraic approaches, and point out general applications of

[1] Spatial Cognition Research Center (SFB/TR 8), University of Bremen, Germany., email: {cschultz, bhatt}@informatik.uni-bremen.de

[2] CLP($\mathcal{QS}$): A Declarative Spatial Reasoning System. www.spatial-reasoning.com

[3] Internally (within CLP($\mathcal{QS}$)), all reasoning tasks are reduced to a polynomial encoding of spatial relations such that constraint solving can be applied by the constraint logic programming engine in its query-answering process.

[4] The inability of relational algebraic methods to handle containment problems is illustrated in [Ladkin and Maddux, 1988]; our approach is able to handle such problems, but we leave out this aspect in the paper because of space restrictions.

our approach in a range of spatial information systems and cognitive technologies involving spatio-linguistic abstraction and computation.

## 2 Logic Programming with Spatial Relations

Spatial information consists of *objects* and *relations* between objects. This is expressed as a constraint network $G = (N, E)$, where the nodes $N$ of the network are spatial *objects* and the edges between nodes specify the relations between the objects. An object belongs to given object *domain*, e.g. points, lines, squares, and circles in 2D Euclidean space, and cuboids, vertically-extruded polygons, spheres, and cylinders in 3D Euclidean space. We denote the object domain of node $i$ as $U_i$. A node may refer to a partially ground, or completely geometrically ground object, such that $U_i$ can be a proper subset of the full domain of that object type. Each element $i' \in U_i$ is called an *instance* of that object domain. A *configuration* of objects is a set of instances $\{i'_1, \dots, i'_n\}$ of nodes $i_1, \dots, i_n$ respectively. A binary relation $R_{ij}$ between nodes $i, j$ distinguishes a set of relative configurations of $i, j$; relation $R_{ij}$ is said to *hold* for those configurations, $R_{ij} \subseteq U_i \times U_j$. In general, an $n$-ary relation for $n \geq 1$ distinguishes a set of configurations between $n$ objects: $R_{i_1,\dots,i_n} \subseteq U_{i_1} \times \dots \times U_{i_n}$. An edge between nodes $i, j$ is assigned a logical formula over relation symbols $R_1, \dots, R_m$ and logical operators $\vee, \wedge, \neg$; given an interpretation $i', j'$, the formula for edge $e$ is interpreted in the standard way, denoted $e(i', j')$:

- $R_1 \equiv (i', j') \in R_{1_{ij}}$
- $(R_1 \vee R_2) \equiv (i', j') \in R_{1_{ij}} \cup R_{2_{ij}}$
- $(R_1 \wedge R_2) \equiv (i', j') \in R_{1_{ij}} \cap R_{2_{ij}}$
- $(\neg R_1) \equiv (i', j') \in (U_i \times U_j) \setminus R_{1_{ij}}$

An edge between $i, j$ is *satisfied* by a configuration $i', j'$ if $e(i', j')$ is *true* (this is generalised to $n$-ary relations). A spatial constraint network $G = (N, E)$ is *consistent* if there exists a configuration of $N$ that satisfies all edges in $E$.

### 2.1 Constraint Logic Programming

We now have a logical framework for talking about spatial objects and relations. In a broader AI context, we can express arbitrary domain rules that can also have a spatial component. For example, we could take the bounding boxes and direction vectors of two people from a video feed and determine whether the people are in conversation by specifying a formula involving spatial relations between the objects and other domain-specific non-spatial aspects.

But how can we evaluate the truth of the spatial relations? One method is to parameterise the objects and encode the relations as polynomial equations and inequalities. For example, we define an axis-aligned *square* as a 2D point $(x, y)$ of its bottom-left corner, and a side length $l$, where $x, y, l$ are reals. Two squares $s_1, s_2$ are *disconnected* if $(s_{1_x} + s_{1_l} < s_{2_x})$ or $(s_{1_y} + s_{1_l} < s_{2_y})$ or the converse inequalities. If the system of polynomial expressions is satisfiable, then the spatial constraint network is consistent; the variables may be assigned to a real value (ground) or not (unground), meaning that we can evaluate any combination of spatial relations between objects, and the objects can be ground, partially ground, or completely unground. Thus, we can integrate spatial reasoning and logic programming using *Constraint Logic Programming* (CLP) [Jaffar and Maher, 1994]; this system is called CLP over qualitative spatial domains, CLP($\mathcal{QS}$). Many linear and non-linear solvers are employed: CLP($\mathbb{R}$), Satisfiability Modulo Theories (SMT), Cylindrical Algebraic Decomposition (CAD), etc.

Notice that the definition of *disconnected* can be satisfied in four mutually exclusive (qualitative) ways: $s_1$ is *left of* $s_2$, $s_1$ is *below* $s_2$, etc. In the context of a larger spatial constraint network, it may be the case that only one of these options is consistent with the other relations in the network. Thus, CLP($\mathcal{QS}$) has two core components: (1) a *search procedure* for identifying important sub-networks, (2) a *transformation strategy* for efficiently solving a corresponding system of polynomials.

### 2.2 Related Work on Spatial Logics

The region connection calculus (RCC) is a spatial logic of topological relations between regions [Randell et al., 1992]. The theory is based on a single *connects* predicate $C(x, y)$ interpreted as the topological closures of regions $x$ and $y$ having at least one point in common (i.e. the regions *touch* at their boundaries or their interiors *overlap*). The authors identify fourteen binary relations that form a subsumption hierarchy, in addition to defining Boolean operators *sum, product, complement, difference*, and a unique *universal* region.

We adopt the language of various RCC relations: *disconnected* (dc), *externally connected* (ec), *partial overlap* (po), *equal* (eq), *part of* (p) *proper part of* (pp), *discrete from* (dr). Let $\mathbf{I}A$ be the interior of $A$:

$$A \ \mathbf{dc} \ B \equiv_{\text{def}} \neg\exists x(x \in A \cap B)$$

$$A \ \mathbf{ec} \ B \equiv_{\text{def}} \exists x(x \in A \cap B) \wedge \neg\exists x(x \in \mathbf{I}A \cap \mathbf{I}B)$$

$$A \ \mathbf{po} \ B \equiv_{\text{def}} \exists x(x \in \mathbf{I}A \cap \mathbf{I}B) \wedge \exists x(x \in \mathbf{I}A \cap \neg B) \wedge \exists x(x \in \neg A \cap \mathbf{I}B)$$

$$A \ \mathbf{eq} \ B \equiv_{\text{def}} \forall x(x \in A \leftrightarrow x \in B)$$

$$A \ \mathbf{dr} \ B \equiv_{\text{def}} A \ \mathbf{dc} \ B \vee A \ \mathbf{ec} \ B$$

$$A \ \mathbf{p} \ B \equiv_{\text{def}} \forall x(x \in A \rightarrow x \in B)$$

$$A \ \mathbf{pp} \ B \equiv_{\text{def}} A \ \mathbf{p} \ B \wedge \neg(A \ \mathbf{eq} \ B)$$

Wolter and Zakharyaschev [2000] extend RCC8 by also allowing Boolean combinations of regions as possible interpretations of a region variable in RCC8; the theory is called BRCC8.

## 3 The *SPLIT* relation

The key mechanism in our framework for defining topological relations between boolean combinations of objects is the $\mathbf{SPLIT}$ relation. Informally, $\mathbf{SPLIT}$ divides a pair of objects into a set of non-overlapping objects that covers the original objects by partitioning them along their *intersection boundaries*. Three additional relations are defined, namely $\mathbf{DIFF}$, $\mathbf{INT}$, and $\mathbf{XOR}$, that select the relevant subset of non-overlapping objects defined by $\mathbf{SPLIT}$ for difference, intersection, and symmetric difference, respectively. Given a spatial domain, the two questions are: (1) does a $\mathbf{SPLIT}$ relation exist? (2) How can the $\mathbf{SPLIT}$ relation be efficiently implemented? The following set of properties for a $\mathbf{SPLIT}$ relation must hold in order for the consistency problem to be decidable in our presented framework.

**Definition 1.** *Let $S$ be a domain of spatial regions in a topological space $(U, \mathbf{I})$. Let $a, b \in S$ and $C \subset S$, then the relation $\mathbf{SPLIT}_S(a, b, C)$ on a domain $S$ holds if:*

1. $|C| \in \mathbb{N}$,
2. $\forall c_1, c_2 \in C(c_1 \neq c_2 \rightarrow c_1 \ \mathbf{dr} \ c_2)$,
3. $a \cup b = \bigcup_{c \in C} c$,
4. $\forall c \in C((c \cap a \setminus b = c) \vee (c \cap b \setminus a = c) \vee (c \cap a \cap b = c))$.

Condition 1 states that $C$ contains a finite number of regions. Condition 2 states that the regions in $C$ must not overlap, although they can touch at their boundaries. Condition 3 states that the unions of the regions in $C$ must be equal to the union of regions $a$ and $b$. Condition 4 states that the regions in $C$ must partition $a$ and $b$ such that each $c$ is either only part of $a$, or only part of $b$, or only part of the intersection of $a$ and $b$. One important property of $\mathbf{SPLIT}_S$ is that $C$ is a subset of $S$. Intuitively, if we think of $\mathbf{SPLIT}_S$ as an operator that takes regions $a$ and $b$ as input and produces $C$ as output, then $S$ is closed under this operation; it divides regions $a$ and $b$ into smaller or equal regions *in the same domain*. This aspect allows us to formulate divide and conquer definitions which are proven to be decidable (Section 3.2).

## 3.1 Splitting Rectangular Polytopes

An axis-aligned rectangular $D$-polytope in Euclidean space is defined as the intersection of two axis-aligned half-spaces in each of the $D$ spatial dimensions, $D > 0$. We can define a split relation $\mathbf{SPLIT}_{\text{DPOLY}}$ by projecting the corresponding half-spaces of the polytope onto a line parallel to the dimension axis, resulting in the intersection of two rays (i.e. 1-D half-spaces) in each dimension. Two such polytopes $a, b$ are therefore defined by four bounding points. If the projections of $a$ and $b$ intersect then the four points $x_1 \leq x_2 < x_3 \leq x_4$ can define between one and three non-overlapping intervals that cover the projections of $a$ and $b$, e.g. $[x_1, x_2], [x_2, x_3], [x_3, x_4]$.

If there exists a dimension where the projections of $a$ and $b$ do not intersect, then the polytopes do not intersect, i.e. $\mathbf{SPLIT}_{\text{DPOLY}}(a, b, \{a, b\})$. Otherwise, a $D$-polytope can be constructed by selecting one of the non-overlapping intervals for each of the $D$ dimensions; the set of polytopes is non-overlapping if each polytope is a unique combination of intervals. Let $C$ be the subset of these polytopes that intersects either $a$ or $b$, or both $a$ and $b$, in the relation $\mathbf{SPLIT}_{\text{DPOLY}}(a, b, C)$.[5] The (finite) number of such polytopes, $|C|$ is in the range $[1, 3^D]$. $C$ is a finite non-overlapping subset of axis-aligned $D$-polytopes that cover $a$ and $b$, therefore $\mathbf{SPLIT}_{\text{DPOLY}}$ satisfies Conditions $1-4$ in Definition 1.

Each derivative relation $\mathbf{DIFF}$, $\mathbf{INT}$, $\mathbf{XOR}$ selects different subsets of $C$ from $\mathbf{SPLIT}$. Let $a, b \in S$, and $C \subset S$. If SPLIT$(a, b, C)$, then:

(1) DIFF$(a, b, \{c \in C | c \cap b = \emptyset\})$,
(2) INT$(a, b, \{c \in C | (c \cap (b \cap a)) = c\})$,
(3) XOR$(a, b, \{c \in C | (c \cap (b \cap a)) = \emptyset\})$.

For example, Figure 1 illustrates the difference operation $\mathbf{DIFF}_{\text{RECT}}$ for rectangles. Using the $\mathbf{SPLIT}$ relation, the problem of determining whether a topological relation between unions of objects can be decided using a divide and conquer approach, where $A$, $B_1$ and $B_2$ can be objects, or unions of objects:

$$A \ \mathbf{dc} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \ A \ \mathbf{dc} \ B_1 \wedge \ A \ \mathbf{dc} \ B_2$$

$$A \ \mathbf{p} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \\ A \ \mathbf{p} \ B_1 \vee \mathbf{DIFF}(A, B_1, A') \wedge \ A' \ \mathbf{p} \ B_2$$

$$A \ \mathbf{pi} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \ B_1 \ \mathbf{p} \ A \wedge \ B_2 \ \mathbf{p} \ A$$

$$A \ \mathbf{eq} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \\ B_1 \ \mathbf{p} \ A \wedge \mathbf{DIFF}(A, B_1, A') \\ \wedge \mathbf{DIFF}(B_2, B_1, B') \wedge \ A' \ \mathbf{eq} \ B'$$

$$A \ \mathbf{pp} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \\ A \ \mathbf{p} \ \textit{union}(B_1, B_2) \wedge \neg(A \ \mathbf{eq} \ \textit{union}(B_1, B_2))$$

$$A \ \mathbf{ppi} \ \textit{union}(B_1, B_2) \ \equiv_{\text{def}} \\ A \ \mathbf{pi} \ \textit{union}(B_1, B_2) \wedge \neg(A \ \mathbf{eq} \ \textit{union}(B_1, B_2))$$

[5] Formally, let the projection of $D$-polytope $s$ onto dimension $d$ be denoted $\pi_d(s)$. Then $\forall c \in C(\forall d \in D(\pi_d(c) \cap \pi_d(a) \neq \emptyset) \vee \forall d \in D(\pi_d(c) \cap \pi_d(b) \neq \emptyset))$.

The operations $\mathbf{DIFF}, \mathbf{INT}$, and $\mathbf{XOR}$ each produce a union of objects, and so we can define the topological relations between other booleans based on these operations and the topological relations for unions above. Let $\mathbf{r}$ be a relation $\mathbf{r} \in \{\mathbf{dc}, \mathbf{pi}, \mathbf{eq}\}$.

$$A \ \mathbf{r} \ \textit{difference}(B_1, B_2) \ \equiv_{\text{def}} \ \mathbf{DIFF}(B_1, B_2, B') \wedge A \ \mathbf{r} \ B' \quad (1)$$

$$A \ \mathbf{r} \ \textit{intersection}(B_1, B_2) \ \equiv_{\text{def}} \ \mathbf{INT}(B_1, B_2, B') \wedge A \ \mathbf{r} \ B' \quad (2)$$

$$A \ \mathbf{r} \ \textit{xor}(B_1, B_2) \ \equiv_{\text{def}} \ \mathbf{XOR}(B_1, B_2, B') \wedge A \ \mathbf{r} \ B' \quad (3)$$

One exception is the relation $\mathbf{pp}$ which in all cases has a more simple definition.

$$A \ \mathbf{pp} \ \textit{difference}(B_1, B_2) \ \equiv_{\text{def}} \ A \ \mathbf{pp} \ B_1 \wedge \ A \ \mathbf{dc} \ B_2 \quad (4)$$

$$A \ \mathbf{pp} \ \textit{intersection}(B_1, B_2) \ \equiv_{\text{def}} \ A \ \mathbf{pp} \ B_1 \wedge \ A \ \mathbf{pp} \ B_2 \quad (5)$$

$$A \ \mathbf{pp} \ \textit{xor}(B_1, B_2) \ \equiv_{\text{def}} \ (A \ \mathbf{pp} \ B_1 \vee A \ \mathbf{pp} \ B_2) \wedge \neg(A \ \mathbf{pp} \ B_1 \wedge A \ \mathbf{pp} \ B_2) \quad (6)$$

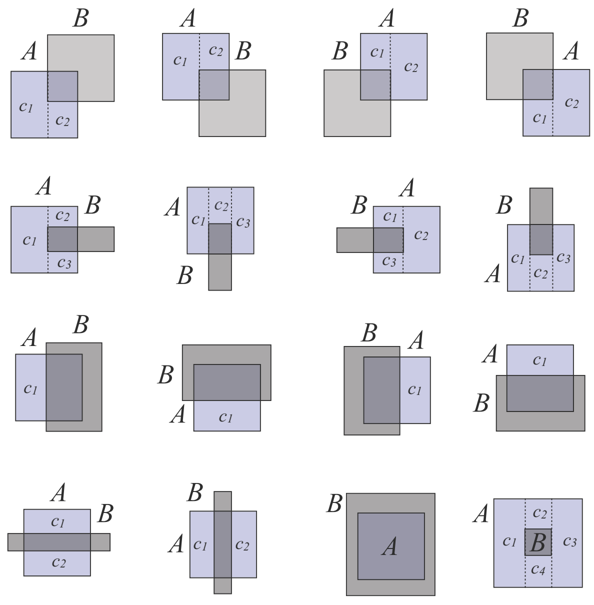


**Figure 1:** Complete set of qualitative cases for the difference of $B$ from $A$; the result is a set of non-overlapping rectangles $C$.

## 3.2 Decidability

$\mathbf{SPLIT}_S$ partitions a pair of $S$ objects into a union of a finite number of non-overlapping $S$ objects. This is extended by Algorithm 1 for partitioning one union by another union.

To prove that each relation is decidable we need to show that recursive calls to the procedure eventually terminate (i.e. repeatedly feeding the output of Algorithm 1 back into Algorithm 1 as the input of a subsequent call reaches a fixpoint).

**Algorithm 1:** Procedure $\mathbf{SPLIT}(A, B, C)$
***Input***: $A = \text{union}(a_1, \ldots, a_n), B = \text{union}(b_1, \ldots, b_m)$
***Output***: $C = \text{union}(c_1, \ldots, c_k)$

```
1 for each $b \in B$
2     $A' = \{\}$
3     for each $a \in A$
4         SPLIT$(a, b, C')$
5         $A' \leftarrow A' \cup C'$
6     $A \leftarrow A'$
7 $C \leftarrow A$
```

**Theorem 1.** *Given constraint network $G = (N, E)$, recursive calls to Algorithm 1 will terminate with initial input $A =$ union$(a_1, \ldots, a_n)$, $B =$ union$(b_1, \ldots, b_m)$, where $A \subseteq N, B \subseteq N$*

*Proof.* As $N$ is finite by definition then $n$ is finite and the first iteration of the inner *for* loop (line 3) will terminate. By the definition of $\mathbf{SPLIT}$, each $C'$ (line 4) is finite and so $A'$ will only increase by a finite amount (line 5). Thus $A$ (line 6) will still be finite. Therefore *all* iterations of the inner loop (line 3) will terminate. As $N$ is finite then $m$ is finite; because the inner loop (line 3) will always terminate and $m$ is finite then the outer loop (line 1) will also terminate. Finally, $|C'| > 2$ (line 4) only if the interior of $a$ and $b$ overlap, but by definition the interior of regions in $C'$ do not overlap (Condition 2).[6] The output of Algorithm 1 is a union of $C'$ (line 5,6,7) thus eventually a fixpoint will be reached (i.e. $\mathbf{SPLIT}(A, B, \{A \cup B\})$. □

## 4 Efficient Reasoning Methods for a Subclass of Packing and Contact Problems

While the method given is decidable, the search space is enormous: the branching factor of Algorithm 1 is 17 from the $\mathbf{SPLIT}$ relation (i.e. the cases in Figure 1 and the case when $a$ and $b$ are discrete), and the depth of the search is between $O(nm)$ (when input $A$ equals output $C$) and the theoretical worst case $O(c^m n)$ (i.e if every occurrence of $\mathbf{SPLIT}$ introduces the maximum number of objects).

The search procedures for spatial reasoning greatly benefit from optimising at a high level of abstraction, rather than pushing the task of optimisation down to the solver level. Depending on the relation and the properties of the objects, significantly more efficient methods are employed. In this section we focus on a particular class of problems where there is a set of objects $B_1, \ldots, B_n$ such that (a) they are completely geometrically undefined, and (b) they have identical constraints (i.e. the objects $B_i$ are *interchangeable* in every consistent instantiation). That is, in a configuration that satisfies the constraint network, the instances for object $B_i$ can be distinct from some other object $B_j$, however, the instantiations for $B_i$ and $B_j$ could be swapped and the new configuration would still satisfy the network.

This problem class corresponds to a range of packing problems (e.g. the union of $B$ is equal to $A$) and contact problem (e.g. all objects in $B$ are discrete from each other, and externally connect to $A$). Figure 2 illustrates a general constraint network with $B_1 - B_4$ that corresponds to this class.

**Restricted $\mathbf{DIFF}$.** Relations such as $A$ $\mathbf{eq}$ *union*$(B_1, B_2)$ require that the union has no gaps. Thus, when trying to arrange $B_1, \ldots, B_n$ to pack rectangle $A$ we can do so incrementally in such a way as to avoid gaps by placing rectangles in the bottom-left-most corner. This skips cases where $B$ has an exposed left edge (i.e. we automatically enforce the constraint $B_{i_x} \leq A_{j_x}$), reducing the possible split cases by a half, as illustrated in Figure 3.

[6] That is, if SPLIT$(a, b, \{c1, c2\})$ then SPLIT$(c1, c2, \{c1, c2\})$.

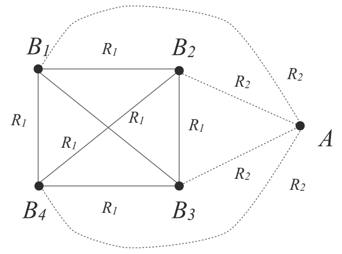


**Figure 2:** An example of a constraint network in the class of packing and contact problems.

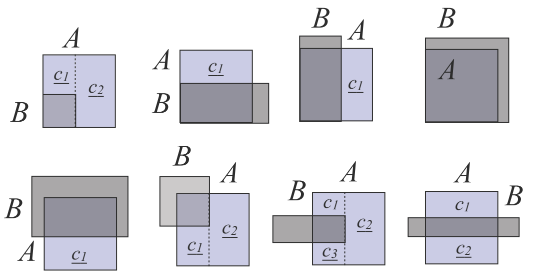


**Figure 3:** Complete set of qualitative cases for restricted $\mathbf{DIFF}$ relation.

**Object Anchoring.** Given a constraint network between objects with no geometric information, if we *ground* certain parameters of one of the objects then we are solving an easier version of the same problem (i.e. less free variables) - this is due to scale invariance from reals being dense. E.g. determining whether four unground circles can mutually touch is the same as determining whether three circles and a fourth completely ground circle can mutually touch. By judiciously selecting the object parameters to ground, it is possible to convert a non-linear problem into a linear problem (see Golden Rectangles problem, Section 4.1), or reduce the runtime complexity of non-linear problems by orders of magnitude.[7]

**Object Interchangeability.** If a set of unground objects have identical constraints, then they are *interchangeable* during the search procedure. Thus, the combination of relations being explored is more relevant than the choice of the object in each relation. E.g. during the $\mathbf{SPLIT}$ procedure, object $B_i$ of union $B$ is used to split a subrectangle $A_j$ of rectangle $A$; if we exhaust all possible ways of $B_i$ splitting region $A_j$ with no success then there is no need to try to split $A_j$ with some other $B_k$ at a later stage, due to interchangeability.

## 4.1 Problem Instances and Empirical Analysis

In this section we present a range of problem instances in the class of packing and contact problems. Tables 1 and 2 present the experiment time results for each problem instance, utilising all pruning methods presented. Experiments were run on a MacBookPro, OS X 10.6.3, 2.66 GHz.

The results clearly show that the pruning methods employed have a significant impact on the runtime performance (without pruning methods, most problems were still not solved after an hour).

**Geometry of Solids Problem** Tarski [1956] defines a geometric *point* using only a language of spheres and the qualitative spatial *between* and *congruence* relations. Borgo [2013] shows that this can be accomplished using a language of hypercubes (for dimension $d \geq$

[7] Solving non-linear polynomials using CAD has doubly-exponential complexity in the number of free variables [Davenport and Heintz, 1988], and thus eliminating three variables by grounding one object can reduce the complexity significantly.

| Problem | Consistent | Time (secs) |
|---|---|---|
| Geometry of Solids | yes ($P_A = P_B$) | 0.08 |
| | no ($P_A \neq P_B$) | 1.52 |
| Rectangle Contact | yes ($n = 4$) | 0.02 |
| | no ($n = 5$) | 1.01 |
| Square Fitting | yes ($n = 4$) | 0.01 |
| | no ($n = 5$) | 3.60 |
| Golden Rectangle | yes (golden($R$)) | 0.05 |
| | no ($\neg$golden($R$)) | 0.11 |

**Table 1:** *Time to solve benchmark problems in Section 4.1.*

| $n$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| $Time(sec)$ | 0.01 | 0.06 | 0.01 | 2.91 | 2.11 | 0.09 | 22.98 | 0.04 |

**Table 2:** *Time to solve the square packing problem for $n = 2 \ldots 9$.*

2) and mereological relations. A simple and less general method uses dimension-dependent qualitative specifications on hypercubes.

The key idea is that a point can be defined by the convergence of a set of regions on that point. Thus, we need to uniquely determine when a pair of hypercubes are concentric using a restricted language of parthood (i.e. *equal, discrete-from, part-of, union*); from this we can construct points and Euclidean geometry.

Let square $A$ be the union of four squares $A1 - A4$ such that $A1 - A4$ are *discrete* from each other; define squares $B$ and $B1 - B4$ similarly. Let $A_i$ be a *proper part* of $B_i$ for $1 \leq i \leq 4$. $A$ and $B$ are necessarily concentric (see Fig. 4).[8] Using CLP($\mathcal{QS}$) we can prove that indeed the definition is sound.

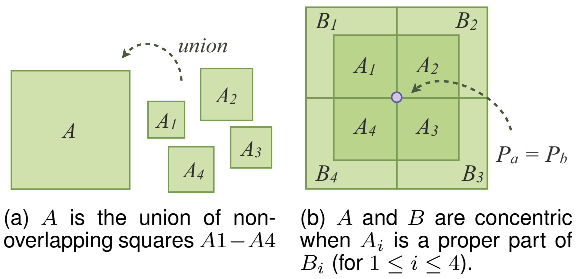
```
?- A=square(_,_), B=square(_,_),
|
|  square_list(4,[A1,A2,A3,A4]),
|  square_list(4,[B1,B2,B3,B4]),
|
|  mereology(rcc5(dr), group([A1,A2,A3,A4])),
|  mereology(rcc5(dr), group([B1,B2,B3,B4])),
|
|  topology(rcc8(eq),A,union(A1,union(A2,union(A3,A4)))),
|  topology(rcc8(eq),B,union(B1,union(B2,union(B3,B4)))),
|
|  mereology(rcc5(pp),A1,B1),mereology(rcc5(pp),A2,B2),
|  mereology(rcc5(pp),A3,B3),mereology(rcc5(pp),A4,B4),
|
|  centre(A,Pa),
|  centre(B,Pb),
|  topology(equal, Pa,Pb).
true.
|  ...
|  topology(not_equal, Pa,Pb).
false.
```

A subproblem is determining whether a square can be packed with $n$ smaller squares (of any size). For $2 \leq n \leq 9$ CLP($\mathcal{QS}$) determines that $n \in \{2, 3, 5\}$ has no solution and the rest do, as illustrated in Figure 5.

**Contact Problems** Certain tasks require combining size and topological relations. Standard approaches to QSR employ algebraic closure by ensuring that all sub-graphs with 3 vertices are satisfiable. Thus, any problem that inherently requires checking four or more objects simultaneously is beyond algebraic closure. A simple example is fitting a set of same-sized squares around a smaller square, as illustrated in Figure 6. We can solve these problems with our encoding.

```
?- A=square(_,_),square_list(5, Sqrs),
|    size(equal, group(Sqrs),
|    size(smaller,A,group(Sqrs)),
|
|    topology(rcc8(ec), A, group(Sqrs) ),
|    mereology(rcc5(dr), group(Sqrs) ).
false.
```

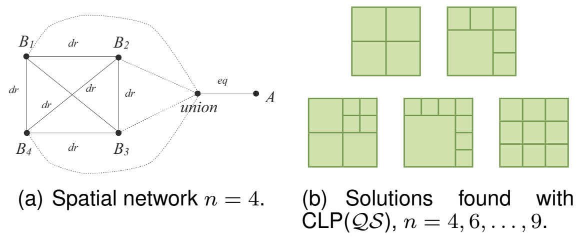
Another interesting contact problem is determining whether a num-



(a) $A$ is the union of non-overlapping squares $A1 - A4$

(b) $A$ and $B$ are concentric when $A_i$ is a proper part of $B_i$ (for $1 \leq i \leq 4$).

**Figure 4:** Characterising concentric squares using mereology.



(a) Spatial network $n = 4$.

(b) Solutions found with CLP($\mathcal{QS}$), $n = 4, 6, \ldots, 9$.

**Figure 5:** $n$ Square packing problem.

ber of objects can be mutually externally connected. CLP($\mathcal{QS}$) solves this $n$ contact problem for rectangles (up to 4) and circles (up to 4).

**Surfaces in Product Design** The task is to arrange two rectangular sheets $A$, $B$ to cover a sensitive region $C$. The task only provides qualitative information as the product is in the design phase. The region $C$ is larger than each surface, and we need to determine whether they can be combined to cover $C$, as long as $A$ and $B$ are not disconnected.

```
?- A=rectangle(_,_,_),B=rectangle(_,_,_),
|    C=rectangle(_,_,_),
|    size(bigger, C,A),size(bigger, C,B),
|    mereology(rcc5(p), C, union(A,B)).
true.
|    ...
|    topology(rcc8(dc), A, B).
|    mereology(rcc5(p), C, union(A,B)).
false.
```

**Constructive Proofs: The Case of Golden Rectangles**

Rectangle $R$ is *golden* if the ratio of the side lengths $R_w$, $R_h$ is the *golden ratio*. Let $a = \max(R_w, R_h), b = \min(R_w, R_h)$, then

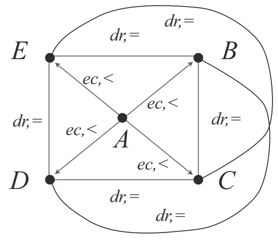
$$\text{golden}(R) \equiv \frac{a+b}{a} = \frac{a}{b} = \frac{1+\sqrt{5}}{2}$$

Golden rectangles have the property that, if a square with a side length $a$ (i.e. equal to the longest side of the rectangle) is placed against the long edge of the rectangle, then their union is also a golden rectangle; no other rectangles have this property.[9]

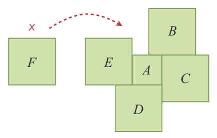
```
?- A=rectangle(_,Aw,Ah),
|    B=rectangle(_,Aw,Aw),
|    golden(A),
|    topology(rcc8(ec),A,B),
|    R=rectangle(_,_,_),
|    topology(rcc8(eq),R,union(A,B)),
|    golden(R).
true.
|    ...
|    size(bigger, value(Aw), value(Ah)),
|    not_golden(R).
false.
```

[8] The key is that, if the union of four squares $A1 - A4$ is itself a square $A$, then $A1 - A4$ must necessarily have the same side length, and they touch at the centre of $A$.

[9] If we do not force the square $B$ to have a side length of $\max(A_w, A_h)$ then CLP($\mathcal{QS}$) finds a solution where the union of $A$ and $B$ is a non-golden rectangle, as illustrated in Figure 7.

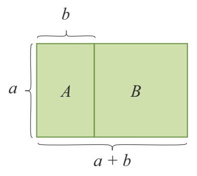


(a) Spatial network $n = 4$

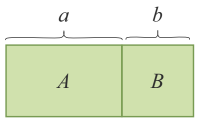


(b) Square fitting problem $n = 5$

**Figure 6:** $n$ Square fitting problem where a maximum of four non-overlapping, same-sized squares can touch a single smaller square.



(a) Both $A$ and $A \cup B$ are golden rectangles.



(b) $A$ is a golden rectangle, but $A \cup B$ is not.

**Figure 7**

We can construct a golden rectangle as follows (see Figure 8): (1) draw a square $A$; (2) draw a circle $C$ centred on the midpoint of the lower edge of $A$ such that it intersects the upper right corner of $A$; (3) draw a rectangle $R$ by extending $A$ until the lower right corner intersects $C$; $R$ is golden. Using CLP($\mathcal{QS}$) we can prove that this procedure can *only* create golden rectangles.[10]

```
?- A=square(point(Ax,Ay),L),
|    centre(A, point(Mx,_)),
|    corner_point(upper_right,A,Pa),
|
|    C=circle(point(Mx,Ay),_),
|    topology(on_boundary, Pa,C),
|
|    R=rectangle(point(Ax,Ay),_,L),
|    corner_point(lower_right,R,Pr),
|    topology(on_boundary, Pr,C),
|    golden(R).
true.
|    ...
|    not_golden(R).
false.
```

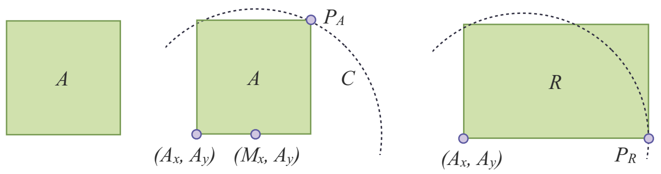


**Figure 8:** Steps for qualitatively constructing a golden rectangle.

## 5 Conclusions

We have presented a method for reasoning about the regions formed from the boolean combination of rectangular polytopes as first-class objects, with a focus of reasoning about mereological and qualitative spatial relations. The developed framework is fully flexible: (**A**). firstly, objects can be of mixed types, different objects can have different spatial dimensions, and objects can be either completely geometrically defined, partially defined, or completely unknown; (**B**). secondly, any combination of qualitative spatial relations can be used (that have been defined within CLP($\mathcal{QS}$)), allowing a mix of topology, distance, shape, orientation etc. Our framework unifies adaptations of a number of related and very well established research problems such as square fitting and tiling (typically these are optimisation problems, whereas we focus on satisfying qualitative spatial relations), and thus we are able to utilise the advanced research results from these areas, particularly pruning methods. We have fully implemented our method in the system CLP($\mathcal{QS}$) with a range of search optimisations such as object anchoring that can reduce the complexity class of a problem (e.g. from solving a system of non-linear to linear polynomials). We have benchmarked our system using a range of classic problems, some of which are well known to be unsolvable using relation algebraic methods for qualitative spatial reasoning, namely, qualitative constructive proofs, packing, and contact problems.

[10] The resulting constraints are non-linear, however, using *object anchoring* CLP($\mathcal{QS}$) reduces this into a linear problem by grounding $A$.

## References

M. Aiello, I. E. Pratt-Hartmann, and J. F. v. Benthem. *Handbook of Spatial Logics*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007. ISBN 978-1-4020-5586-7.

M. Bhatt, H. Guesgen, S. Wölfl, and S. Hazarika. Qualitative spatial and temporal reasoning: Emerging applications, trends, and directions. *Spatial Cognition & Computation*, 11(1):1–14, 2011a.

M. Bhatt, J. H. Lee, and C. Schultz. **CLP(QS)**: A declarative spatial reasoning framework. In *Proceedings of the 10th international conference on Spatial information theory*, COSIT'11, pg. 210–230, Berlin, Heidelberg, 2011b. Springer-Verlag. ISBN 978-3-642-23195-7.

M. Bhatt, C. Schultz, and C. Freksa. The 'Space' in Spatial Assistance Systems: Conception, Formalisation and Computation. *Representing space in cognition: Interrelations of behavior, language, and formal models. Series: Explorations in Language and Space*. 978-0-19-967991-1, Oxford University Press, 2013.

M. Bhatt, C. Schultz, and M. Thosar. Computing narratives of cognitive user experience for building design analysis: KR for industry scale computer-aided architecture design. *Principles of Knowledge Representation and Reasoning: Proc. of 14th Intl. Conf*, 2014. (to appear)

S. Borgo. Euclidean and mereological qualitative spaces: A study of scc and dcc. In C. Boutilier, editor, *IJCAI*, pages 708–713, 2009.

S. Borgo. Spheres, cubes and simplexes in mereogeometry. *Logic and Logical Philosophy*, 22(3):255–293, 2013.

J.-F. Condotta, M. Saade, and G. Ligozat. A generic toolkit for n-ary qualitative temporal and spatial calculi. In *TIME*, pages 78–86. IEEE Computer Society, 2006. ISBN 0-7695-2617-9.

J. H. Davenport and J. Heintz. Real quantifier elimination is doubly exponential. *Journal of Symbolic Computation*, 5(1):29–35, 1988.

J. Jaffar and M. J. Maher. Constraint logic programming: A survey. *The journal of logic programming*, 19:503–581, 1994.

D. Kapur and J. L. Mundy, editors. *Geometric Reasoning*. MIT Press, Cambridge, MA, USA, 1988. ISBN 0-262-61058-2.

H. Guesgen. Spatial reasoning based on Allen's temporal logic. Technical Report TR-89-049, ICSI, Berkeley, California, 1989.

R. Kontchakov, Y. Nenov, I. Pratt-Hartmann, and M. Zakharyaschev. Topological logics with connectedness over euclidean spaces. *ACM Trans. Comput. Log.*, 14(2):13, 2013.

P. B. Ladkin and R. D. Maddux. On binary constraint networks. Technical report, 1988.

G. Ligozat. *Qualitative Spatial and Temporal Reasoning*. Wiley-ISTE, 2011.

D. A. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In *KR'92. Principles of Knowledge Representation and Reasoning*, pages 165–176. Morgan Kaufmann, San Mateo, California, 1992.

J. Renz and B. Nebel. Qualitative spatial reasoning using constraint calculi. In *Handbook of Spatial Logics*, pages 161–215. 2007.

C. Schultz and M. Bhatt. Towards a declarative spatial reasoning system. In *ECAI 2012*, pg. 925–926, 2012.

A. Tarski. A general theorem concerning primitive notions of euclidean geometry. *Indagationes Mathematicae*, 18(468):74, 1956.

F. Wolter and M. Zakharyaschev. Spatial representation and reasoning in RCC-8 with boolean region terms. In *ECAI 2000*, pg. 244–248. 2000.

H. Simonis and B. O'Sullivan. Search strategies for rectangle packing. In *Constraint Programming*, pg. 52–66. Springer, 2008.