Spatial Symmetry Driven Pruning Strategies for Efficient Declarative Spatial Reasoning

Carl Schultz[1,3(✉)] and Mehul Bhatt[2,3]

[1] Institute for Geoinformatics, University of Münster, Münster, Germany
schultzc@uni-muenster.de
[2] Department of Computer Science, University of Bremen, Bremen, Germany
[3] The DesignSpace Group, Bremen, Germany

Abstract. Declarative spatial reasoning denotes the ability to (declaratively) specify and solve real-world problems related to geometric and qualitative spatial representation and reasoning within standard knowledge representation and reasoning (KR) based methods (e.g., logic programming and derivatives). One approach for encoding the semantics of spatial relations within a declarative programming framework is by systems of polynomial constraints. However, solving such constraints is computationally intractable in general (i.e. the theory of real-closed fields).

We present a new algorithm, implemented within the declarative spatial reasoning system CLP(QS), that drastically improves the performance of deciding the consistency of spatial constraint graphs over conventional polynomial encodings. We develop pruning strategies founded on spatial symmetries that form equivalence classes (based on affine transformations) at the qualitative spatial level. Moreover, pruning strategies are themselves formalised as knowledge about the properties of space and spatial symmetries. We evaluate our algorithm using a range of benchmarks in the class of contact problems, and proofs in mereology and geometry. The empirical results show that CLP(QS) with knowledge-based spatial pruning outperforms conventional polynomial encodings by orders of magnitude, and can thus be applied to problems that are otherwise unsolvable in practice.

Keywords: Declarative spatial reasoning · Geometric reasoning · Logic programming · Knowledge representation and reasoning

1 Introduction

Knowledge representation and reasoning (KR) about *space* may be formally interpreted within diverse frameworks such as: (a) analytically founded geometric reasoning & constructive (solid) geometry [21,27,29]; (b) relational algebraic semantics of 'qualitative spatial calculi' [24]; and (c) by axiomatically constructed formal systems of mereotopology and mereogeometry [1]. 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 [5].

S.I. Fabrikant et al. (Eds.): COSIT 2015, LNCS 9368, pp. 331–353, 2015.
DOI: 10.1007/978-3-319-23374-1_16

▷ **Declarative Spatial Reasoning.** In the recent years, *declarative spatial reasoning* has been developed as a high-level commonsense spatial reasoning paradigm aimed at (declaratively) specifying and solving real-world problems related to geometric and qualitative spatial representation and reasoning [4]. A particular manifestation of this paradigm is the constraint logic programming based CLP(QS) spatial reasoning system [4,33,34] (Sect. 2).

▷ **Relational Algebraic Qualitative Spatial Reasoning.** The state of the art in qualitative spatial reasoning using relational algebraic methods [24] has resulted in prototypical algorithms and black-box systems that do not integrate with KR languages, such as those dealing with semantics and conceptual knowledge necessary for handling background knowledge, action &change, relational learning, rule-based systems etc. Furthermore, relation algebraic qualitative spatial reasoning (e.g. LR [25]), while efficient, is incomplete in general [22–24].[1] Alternatively, constraint logic programming based systems such as CLP(QS) [4] and others (see [9,10,18,20,28,29]) adopt an analytic geometry approach where spatial relations are encoded as systems of polynomial constraints;[2] while these methods are sound and complete (see Sect. 2.2), they have prohibitive computational complexity, $O(c_1^{c_2^n})$ in the number of polynomial variables n, meaning that even relatively simple problems are not solved within a practical amount of time via "naive" or direct encodings as polynomial constraints, i.e. encodings that lack common-sense knowledge about spatial objects and relations. On the other hand, highly efficient and specialised geometric theorem provers (e.g. [12]) and geometric constraint solvers (e.g. [17,27]) exist. However, these provers exhibit highly specialised and restricted spatial languages[3] and lack (a) the direct integration with more general AI methods and (b) the capacity for incorporating modular common-sense rules about space in an extensible domain- and context-specific manner.

The aims and contributions of the research presented in this paper are two-fold:

1. to further develop a KR-centered declarative spatial reasoning paradigm such that spatial reasoning capabilities are available and accessible within AI

[1] *Incompleteness* refers to the inability of a spatial reasoning method to determine whether a given network of qualitative spatial constraints is consistent or inconsistent in general. Relation-algebraic spatial reasoning (i.e. using algebraic closure based on weak composition) has been shown to be incomplete for a number of spatial languages and cannot guarantee *consistency* in general, e.g. relative directions [23] and containment relations between linearly ordered intervals [22], Theorem 5.9.

[2] We emphasise that this analytic geometry approach that we also adopt is not *qualitative spatial reasoning* in the relation algebraic sense; the foundations are similar (i.e. employing a finite language of spatial relations that are interpreted as infinite sets of configurations, determining consistency in the complete absence of numeric information, and so on) but the methods for determining consistency etc. come from different branches of spatial reasoning.

[3] Standard geometric constraint languages of approaches including [12,17,27] consist of points, lines, circles, ellipses, and coincidence, tangency, perpendicularity, parallelism, and numerical dimension constraints; note the absence of e.g. mereotopology and "common-sense" relative orientation relations [35].

programs and applications areas, and may be seamlessly integrated with other AI methods dealing with representation, reasoning, and learning about non-spatial aspects
2. to demonstrate that in spite of high computational complexity in a general and domain-independent case, the power of analytic geometric—in particular polynomial systems for encoding the semantics of spatial relations – can be exploited by systematically utilising commonsense knowledge about spatial object and relationships at the qualitative level.

We present a new algorithm that drastically improves analytic spatial reasoning performance within KR-based declarative spatial reasoning approaches by identifying and pruning spatial symmetries that form equivalence classes (based on affine transformations) at the qualitative spatial level. By exploiting symmetries our approach utilises powerful underlying, but computationally expensive, polynomial solvers in a significantly more effective manner. Our algorithm is simple to implement, and enables spatial reasoners to solve problems that are otherwise unsolvable using analytic or relation algebraic methods. We emphasise that our approach is independent of any particular polynomial constraint solver; it can be similarly applied over a range of solvers such as CLP(R), SMTs, and specialised geometric constraint solvers that have been integrated into a KR framework.

In addition to AI/commonsense reasoning applications areas such as design, GIS, vision, robotics [3,5–7], we also address application into automating support for proving the validity of theorems in mereotopology, orientation, shape, etc. (e.g. [8,36]). Building on such foundational capabilities, another outreach is in the area of computer-aided learning systems in mathematics (e.g. at a high-school level). For instance, consider Proposition 9, Book I of Euclid's *Elements*, where the task is to bisect an angle using only an unmarked ruler and collapsable compass. Once a student has developed what they believe to be a constructive proof, they can employ declarative spatial reasoners to formally verify that their construction applies to all possible inputs (i.e. all possible angles) and manipulate an interactive sketch that maintains the specified spatial relations (i.e. dynamic geometry [17]). A further area of interest is verifying the entries of composition tables that are used in relation algebraic qualitative spatial reasoning [30]: given spatial objects $a, b, c \in U$, composition "look up" tables are indexed by pairs of (base) relations $R_{1_{ab}}$, $R_{2_{bc}}$ and return disjunctions of possible (base) relations $R_{3_{ac}}$. For each entry, the task is to prove $\exists a, b, c \in U\big(R_{1_{ab}} \wedge R_{2_{bc}} \wedge R_{3_{ac}}\big)$ for only those base relations R_3 in the entry's disjunction.

2 Declarative Spatial Reasoning with CLP(QS)

Declarative spatial reasoning denotes the ability of declarative programming frameworks in AI to handle *spatial objects* and the *spatial relationships* amongst them as *native* entities, e.g., as is possible with concrete domains of Integers, Reals and Inequality relationships. The objective is to enable points, oriented points, directed line-segments, regions, and topological and orientation relationships amongst them as *first-class* entities within declarative frameworks in AI [4].

2.1 Examples of Declarative Spatial Reasoning with CLP(QS)

With a focus on spatial question-answering, the CLP(QS) spatial reasoning system [4,33,34] provides a practical manifestation of certain aspects of the declarative spatial reasoning paradigm in the context of constraint logic programming (CLP).[4] CLP(QS) utilises a high-level language of spatial relations and commonsense knowledge about how various spatial domains behave. Such relations describe sets of object configurations, i.e. qualitative spatial relations such as *coincident*, *left of*, or *partially overlapping*. Through this deep integration of spatial reasoning with KR-based frameworks, the long-term research agenda is to seamlessly provide spatial reasoning in other AI tasks such as planning, non-monotonic reasoning, and ontological reasoning [4]. What follows is a brief illustration of the spatial Q/A capability supported by CLP(QS).

EXAMPLE A. *Massachusetts Comprehensive Assessment System (MCAS).*

Grade 3 Mathematics (2009), Question 12. *Put a square and two right-angled triangles together to make a rectangle. (1) Put the shapes* T_1, T_2, S *illustrated in Fig.* 1*(d) together to make a rectangle. (2) Put the shapes* T_1, T_2, S *in Fig.* 1*(d) together to make a quadrilateral that is not a rectangle.*

CLP(QS) represents right-angle triangles as illustrated in Fig. 1(b). Figure 1(a) and (c) present the CLP(QS) solutions.

Grade 3 Mathematics (2013), Question 17. *(1) How many copies of* T_1 *illustrated in Fig.* 1*(d) are needed to completely fill the region R illustrated in Fig.* 2*(a) without any of them overlapping?*

As presented in Fig. 2, CLP(QS) solves both the geometric definition and a variation where the dimensions of the rectangle and triangles are not given.

EXAMPLE B. *Qualitative Spatial Reasoning with Complete Unknowns.*

In this example CLP(QS) reasons about spatial objects based solely on given qualitative spatial relations, i.e. without any geometric information.

Define three cubes A, B, C*. Put* B *inside* A*, and make* B *disconnected from* C*. What spatial relations can possibly hold between* A *and* C*?*

CLP(QS) determines that A must be *disconnected* from C and provides the inferred corresponding geometric constraints, as illustrated in Fig. 3.

2.2 Analytical Geometry Foundations for Declarative Spatial Reasoning

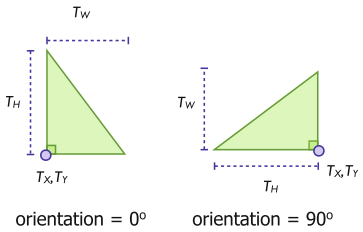
Analytic geometry methods parameterise classes of objects and encode spatial relations as systems of polynomial equations and inequalities [12]. For example, we can define a sphere as having a 3D centroid point (x, y, z) and a radius r, where x, y, z, r are reals. Two spheres s_1, s_2 *externally connect* or *touch* if

$$(x_{s_1} - x_{s_2})^2 + (y_{s_1} - y_{s_2})^2 + (z_{s_1} - z_{s_2})^2 = (r_{s_1} + r_{s_2})^2 \tag{1}$$

[4] Spatial Reasoning (CLP(QS)). www.spatial-reasoning.com.

```
?- T1  = right_triangle(_,_,W,W),
|    T2  = right_triangle(_,_,W,W),
|    S = rectangle(_,W,W),
|    Solution = rectangle(_,_,_,_),
|    topology(rcc8(eq), Solution, tile_union([T1,T2,S])),
|    ground_object(Solution).

T1 = right_triangle(orientation(0), point(0, 0), 5, 5),
W = 5,
T2 = right_triangle(orientation(180), point(5, 5), 5, 5),
S = rectangle(point(5, 0), 5, 5),
Solution = rectangle(point(0, 0), 10, 5) .
```

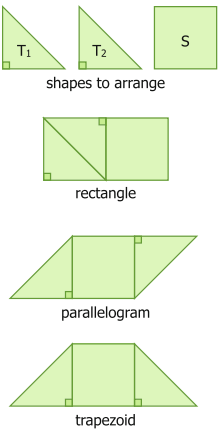
(a) CLP(QS) solution to arranging T_1, T_2, S to form rectangles.



(b) Representing right-angle triangle T in CLP(QS).

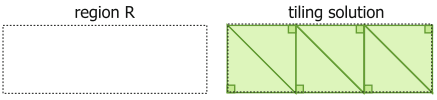
```
?- T1  = right_triangle(_,_,W,W),
|    T2  = right_triangle(_,_,W,W),
|    S = rectangle(_,W,W),
|    topology(rcc8(eq), Solution, tile_union([T1,T2,S])),
|    quadrilateral(Solution),
|    not(Solution = rectangle(_,_,_,_)),
|    ground_object(Solution).

T1 = right_triangle(orientation(90), point(5, 0), 5, 5),
W = 5,
T2 = right_triangle(orientation(270), point(10, 5), 5, 5),
S = rectangle(point(5, 0), 5, 5),
Solution = parallelogram(orientation(0), point(0,0), 10, 5);
...
T1 = right_triangle(orientation(90), point(5, 0), 5, 5),
W = 5,
T2 = right_triangle(orientation(0), point(10, 0), 5, 5),
S = rectangle(point(5, 0), 5, 5),
Solution = trapezoid(orientation(0), point(0, 0), 15, 5, 5);
...
false.
```

(c) CLP(QS) solution to arranging T_1, T_2, S to form non-rectangular quadrilaterals.



(d) Shapes T_1, T_2, S to be arranged and CLP(QS) solutions.

Fig. 1. Using CLP(QS) to solve MCAS Grade 3 Mathematics Test questions (2009).




```
?- between(1,8,TriangleCount),
|    right_triangle_list(TriangleCount,[T1|TRest]),
|    T1 = right_triangle(_,_,1,1),
|    size(equal, T1, TRest),
|    topology(rcc8(eq), rectangle(point(0,0),3,1),
|                  tile_union([T1|TRest])).

TriangleCount = 6
```

(a) CLP(QS) solution to tiling a rectangular region with triangle T_1.

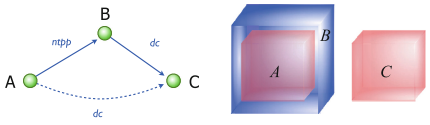
```
?- between(1,8,TriangleCount),
|    right_triangle_list(TriangleCount,Triangles),
|    topology(rcc8(eq), rectangle(_,_,_,_),
|                  tile_union(Triangles)).

TriangleCount = 2;...
TriangleCount = 4;...
TriangleCount = 6;...
TriangleCount = 8;
false.
```

(b) When no geometric information is given, CLP(QS) determines that the number of right-angle triangles must be even.

Fig. 2. Using CLP(QS) to solve MCAS Grade 3 Mathematics Test questions (2013).

```
?- A=cube(_, _), B=cube(_, _), C=cube(_, _),
|    topology(rcc8(ntpp), A, B),
|    topology(rcc8(dc), B, C),
|    topology(Relation, A,C),
|    constraints([[A,'A'),(B,'B'),(C,'C')],Cons),
|    pretty_print_constraints(Cons).

---------------------------------
 CLP(QS) Constraints:

        (1)        B.length+B.x-C.x< -0.0
        (2)        A.z-B.z>0.0
        (3)        A.length-B.length+A.z-B.z<0.0
        (4)        A.y-B.y>0.0
        (5)        A.length-B.length+A.y-B.y<0.0
        (6)        A.x-B.x>0.0
        (7)        A.length-B.length+A.x-B.x<0.0
        (8)        A.length>0.0
        (9)        C.length>0.0

---------------------------------
...
Relation = rcc8(dc),
...
```

(a) CLP(QS) solution and its corresponding inferred geometric constraints.



(1) $B_x + B_l < C_x$	*B* is left of *C*
(2) $A_z > B_z$	*the front face of A* is behind *the front face of B*
(3) $A_z + A_l < B_z + B_l$	*the back face of A* is in front of *the back face of B*
(4) $A_y > B_y$	*the base of A* is above *the base of B*
(5) $A_y + A_l < B_y + B_l$	*the top of A* is below *the top of B*
(6) $A_x > B_x$	*the left side of A* is right of *the left side of B*
(7) $A_x + A_l < B_x + B_l$	*the right side of A* is left of *the right side of B*
(8) $A_l > 0$	*A* side length is positive
(9) $C_l > 0$	*C* side length is positive

(b) Geometric inequalities provided by CLP(QS) that correspond to the qualitative constraints.

Fig. 3. Spatial reasoning about cubes A, B, C with complete geometric unknowns.

If the system of polynomial constraints is satisfiable then the spatial constraint graph is consistent. Specifically, the system of polynomial (in)equalities over variables X is satisfiable if there exists a real number assignment for each $x \in X$ such that the (in)equalities are *true*. Partial geometric information (i.e. a combination of numerical and qualitative spatial information) is utilised by assigning the given real numerical values to the corresponding object parameters. Thus, we can integrate spatial reasoning and logic programming using *Constraint Logic Programming* (CLP) [19]; this system is called CLP over qualitative spatial domains. CLP($\mathcal{QS}$), provides a suitable framework for expressing and proving first-order spatial theorems.

Cylindrical Algebraic Decomposition (CAD) [13] is a prominent sound and complete algorithm for deciding satisfiability of a general system of polynomial constraints over reals and has time complexity $O(c_1^{c_2^n})$ in the number of free variables [2]. Thus, a key focus within analytic spatial reasoning has been methods for managing this inherent intractibility.[5] More efficient refinements of the original CAD algorithm include partial CAD [14]. Symbolic methods for solving systems of multivariate *equations* include the Gröbner basis method [11] and Wu's characteristic set method [40]. In the QUAD-CLP(R) system, the authors improve solving performance by using linear approximations of quadratic constraints and by

[5] Important factors in determining the applicability of various analytic approaches are the degree of the polynomials (particularly the distinction between linear and non-linear) and whether both equality and inequalities are permitted in the constraints.

identifying geometric equivalence classes [28]. Ratschan employs pruning methods, also at the polynomial level, in the *rsolve* system [31,32].

Constructive and iterative (i.e. Newton and Quasi-Newton iteration) methods solve spatial reasoning problems by "building" a solution, i.e. by finding a configuration that satisfies the given constraints [27]. If a solution is found, then the solution itself is the proof that the system is consistent – but what if a solution is not found within a given time frame? In general these methods are incomplete for spatial reasoning problems encoded as nonlinear equations and inequalities of arbitrary degree.[6]

3 Spatial Symmetries

Information about *objects* and their *spatial relations* is formally expressed as a constraint graph $G = (N, E)$, where the nodes N of the graph are spatial objects and the edges between nodes specify the relations between the objects. Objects belong to a *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 (spatial domains are typically infinite). 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 $\{i'_1, \ldots, i'_n\}$ of nodes $i_1, \ldots, i_n$ respectively.

A binary relation R_{ij} between nodes i, j distinguishes a set of relative configurations of i, j; relation R 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: $R_{i_1,\ldots,i_n} \subseteq U_{i_1} \times \cdots \times U_{i_n}$.

An edge between nodes i, j is assigned a logical formula over relation symbols $R_1, \ldots, R_m$ 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 graph $G = (N, E)$ is *consistent* or *satisfiable* if there exists a configuration s of N that satisfies all edges in E, denoted $G(s)$; this is referred to as the *consistency task*. Graph G' is a *consequence* of, or *implied* by, G if every spatial configuration that satisfies G also satisfies G'. This is the *sufficiency task* (or *entailment*) that we commonly apply to constructive proofs, where the task is to prove that objects and relations in G are sufficient for ensuring that particular properties hold in G'.

[6] That is, constructive methods may fail in building a consistent solution, and iterative root finding methods may fail to converge.

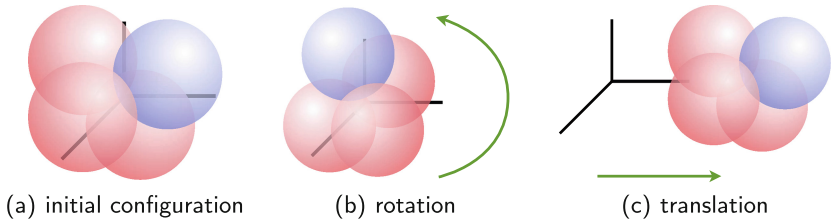


Fig. 4. Topological relations between four spheres maintained after various affine transformations.

Given graph G, two key questions are (1) how to give meaning, or interpret, the spatial relations in G, and (2) how to efficiently determine consistency and produce instantiations of G. That is, we need to adopt a method for *spatial reasoning*.

3.1 An Example of the Basic Concept

A key insight is that spatial configurations form equivalence classes over qualitative relationships based on certain affine transformations. For example, consider the spatial task of determining whether five same-sized spheres can be mutually touching. Suppose we are given a specific numerically defined configuration of four mutually touching spheres as illustrated in Fig. 4(a), and we prove that it is impossible to add an additional mutually touching sphere to this configuration. That is, let $s_1, \ldots, s_4$ be unit spheres (radius 1), centred on points $p_1 = (0, 0, 0)$, $p_2 = (2, 0, 0)$, $p_3 = (1, \sqrt{3}, 0)$, $p_4 = (1, \sqrt{\frac{1}{3}}, \sqrt{\frac{8}{3}})$, respectively. According to Eq. 1, $s_1, \ldots, s_4$ are mutually touching. We prove that a fifth same-sized, mutually touching sphere cannot be added to this configuration by determining that the corresponding system of polynomial constraints is unsatisfiable (the system consists of four constraints with three free variables $x_{s_5}, y_{s_5}, z_{s_5}$, by reapplying Eq. 1 between s_5 and each other sphere, e.g. s_1 *touches* s_5 is $x_{s_5}^2 + y_{s_5}^2 + z_{s_5}^2 = 4$).

Now consider that we apply an affine transformation to the original configuration such as rotation, translation, scaling, or reflection, as illustrated in Fig. 4(b) and (c). After having applied the transformation, it is still impossible to add a fifth mutually touching sphere, because the relevant qualitative (topological) relations are *preserved* under these transformations. Thus, when we proved that it was impossible to add a fifth same-sized mutually touching sphere to the original given configuration, in fact we proved it for a *class* of configurations, specifically, the class of configurations that can be reached by applying an affine transformation to the original configuration. Now, when determining consistency of graphs of qualitative spatial relations, we are not given any specific spatial configurations to work with (i.e. complete absence of numerical information), and instead need to prove consistency over all possible configurations.

The key is that, each time we ground and constrain variables, we are eliminating a *spatial symmetry* from our partially defined configuration. If we

maintain knowledge about symmetries that certain object types have (e.g. spheres have complete rotational symmetry) then we can judiciously "trade" symmetries for unbound variables in our polynomial encoding at a purely symbolic level. Importantly, rather than having to compute symmetries or undertake any complex symmetry detection procedure, we are instead *building knowledge about space and spatial properties of objects into the spatial solver at a declarative level*. Thus, we are able to efficiently reason over an infinite set of possible configurations by incrementally pruning spatial symmetries based on commonsense knowledge about space, and this pruning is exploited by eliminating and constraining variables in the underlying polynomial encoding.

3.2 Theoretical Foundations for Symmetries

Due to the parameterisation of objects, spatial configurations are embedded in n-dimensional Euclidean space $\mathbb{R}^n$ $(1 \leq n \leq 3)$ with a fixed origin point. Let V, W be Euclidean spaces in $\mathbb{R}^n$, each with an origin. Given vectors x, y and constant k, a linear transformation f is a mapping $V \rightarrow W$ such that

$$\begin{aligned} f(x+y) &= f(x) + f(y) \quad (additive) \\ f(kx) &= kf(x) \qquad\quad (homogeneous) \end{aligned}$$

An affine transformation f' is a linear transformation composed with a translation. It is convenient to represent a linear transformation on vector x as a left multiplication of a $d \times d$ real matrix Q, and translation as an addition of vector t, $f'(x) = Qx + t$. We denote a transformation T applied to a spatial configuration of objects s as Ts.

We distinguish particular classes of transformations with respect to the qualitative spatial relationships that are preserved, for example, in $\mathbb{R}^2$ the following matrices represent rotation by θ, uniform scaling by $k > 0$, and horizontal reflection, respectively:

$$\begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}, \begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix}, \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Given transformation T we annotate it with its type $c \in C$, e.g. $C = \{\text{translate, rotate, scale, reflect}\}$ as T^c. Each spatial relation R belongs to a class of relations in Rel, such as *topology, mereology, coincidence, relative orientation, distance*. Let Sym be a function $\text{Sym} : \text{Rel} \rightarrow 2^C$ that represents the classes of transformations that preserve a given class of spatial relations. The Sym function is our mechanism for building knowledge about spatial symmetries into the spatial reasoning system. Let Rel_G be the set of classes of the spatial relations that are used in the spatial constraint graph G, and let $\text{Sym}_G = \bigcap_{R \in \text{Rel}_G} \text{Sym}(R)$.

The following formal Condition on Sym_G states that transformations (applied to the embedding space) define equivalence classes of configurations with respect to the consistency of spatial constraint graphs. When satisfied, this condition provides a theoretically sound foundation for symmetry pruning.

Condition 1. *Given spatial constraint graph G, configuration s, and affine transformation T^c with $c \in Sym_G$ then $G(s)$ is true if and only if $G(T^c s)$.*

Table 1. Polynomial encodings of qualitative spatial relations.

Relation	Polynomial Encoding
Left of (point p, segment s_{ab})	$(x_b - x_a)(y_p - y_a) > (y_b - y_a)(x_p - x_a)$
Collinear (point p, segment s_{ab})	$(x_b - x_a)(y_p - y_a) = (y_b - y_a)(x_p - x_a)$
Right or collinear (point p, segment s_{ab})	$(x_b - x_a)(y_p - y_a) \leq (y_b - y_a)(x_p - x_a)$
Parallel (segments s_{ab}, s_{cd})	$(y_b - y_a)(x_d - x_c) = (y_d - y_c)(x_b - x_a)$
Coincident (point p, segment s_{ab})	$\text{collinear}(p, s_{ab}) \wedge x_p \in [x_a, x_b] \wedge y_p \in [y_a, y_b]$
Coincident (point p, circle c)	$(x_c - x_p)^2 + (y_c - y_p)^2 = r_c^2$
Inside (point p, rectangle a)	$(0 < (p - p_{1_a}) \cdot v_a < w_a) \wedge$ $(0 < (p - p_{1_a}) \cdot v'_a < h_a)$
Intersects (point p, rectangle a)	$(0 \leq (p - p_{1_a}) \cdot v_a \leq w_a) \wedge$ $(0 \leq (p - p_{1_a}) \cdot v'_a \leq h_a)$
Boundary (point p, rectangle a)	$\text{intersects}(p, a) \wedge \neg\, \text{inside}(p, a)$
Outside (point p, rectangle a)	$\neg\, \text{intersects}(p, a)$
Concentric (rectangles a, b)	$\frac{1}{2}(p_{3_a} - p_{1_a}) + p_{1_a} = \frac{1}{2}(p_{3_b} - p_{1_b}) + p_{1_b}$
Part of (rectangles a, b)	$\bigwedge_{i=1\ldots4} \text{intersects}(p_{i_a}, b)$
Proper part (rectangles a, b)	$\neg\, \text{equals}(a, b) \wedge \text{part\_of}(a, b)$
Boundary part of (rectangles a, b)	$\bigwedge_{i=1\ldots4} \text{boundary}(p_{i_a}, b)$
Discrete from (rectangles a, b)	$\bigvee_{i=1\ldots4} \big(\ \text{right\_or\_collinear}(a, (p_{i_b}, p_{(i+1)_b})) \vee$ $\text{right\_or\_collinear}(b, (p_{i_a}, p_{(i+1)_a}))\big)$
Partially overlaps (rectangles a, b)	$\exists p_i \in \mathbb{R}^2 \big(\text{inside}(p_i, a) \wedge \text{inside}(p_i, b)\big) \wedge$ $\exists p_j \in \mathbb{R}^2 \big(\text{inside}(p_j, a) \wedge \text{outside}(p_j, b)\big) \wedge$ $\exists p_k \in \mathbb{R}^2 \big(\text{outside}(p_k, a) \wedge \text{inside}(p_k, b)\big)$

3.3 Polynomial Encodings for Spatial Relations

In this section we define a range of spatial domains and spatial relations with the corresponding polynomial encodings. While our method is applicable to a wide range of 2D and 3D spatial objects and qualitative relations, for brevity and clarity we primarily focus on a 2D spatial domain. Our method is readily applicable to other 2D and 3D spatial domains and qualitative relations, for example, as defined in [4,9,10,28,29,33,34].

- a *point* is a pair of reals x, y
- a *line segment* is a pair of end points p_1, p_2 $(p_1 \neq p_2)$
- a *rectangle* is a point p representing the bottom left corner, a unit direction vector v defining the orientation of the base of the rectangle, and a real width and height w, h $(0 < w, 0 < h)$; we can refer to the vertices of the rectangle: let $v' = (-y_v, x_v)$ be v rotated 90^o counter-clockwise, then $p_1 = p = p_5, p_2 = wv + p_1, p_3 = wv + hv' + p_1, p_4 = hv' + p_1$
- a *circle* is a centre point p and a real radius r $(0 < r)$.

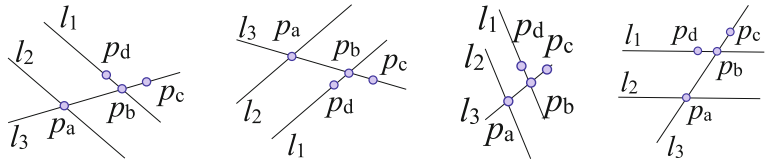


Fig. 5. Affine transformations preserve point coincidence, parallelism, and ratios of distances along parallel lines.

We consider the following spatial relations:

Relative Orientation. Left, right, collinear orientation relations between points and segments, and parallel, perpendicular relations between segments [23].

Coincidence. Intersection between a point and a line, and a point and the boundary of a circle. Also whether the point is in the interior, outside or on the boundary of a region.

Mereology. Part-whole relations between regions [37].

Table 1 presents the corresponding polynomial encodings. Given three real variables v, i, j, let:

$$v \in [i, j] \equiv i \leq v \leq j \vee j \leq v \leq i.$$

Determining whether a point is inside a rectangle is based on vector projection. Point p is projected onto vector v by taking the dot product,

$$(x_p, y_p) \cdot (x_v, y_v) = x_p x_v + y_p y_v$$

Given point a and rectangle b, we translate the point such that the bottom left corner of b is at the origin, project p_a on the base and side vectors of b, and check whether the projection lies within the width and height of the rectangle,

$$0 < (p_a - p_{1_b}) \cdot v_b < w_b$$

$$0 < (p_a - p_{1_b}) \cdot v'_b < h_b$$

Convex regions a, b are disconnected iff there is a hyperplane of separation, i.e. there exists a line l such that a and b lie on different sides of l. This is the basis for determining the *discrete from* relation between rectangles.

3.4 Formalising Knowledge About Symmetries

In this section we formally determine the qualitative spatial relations that are preserved by various affine transformations. A fundamental property of affine transformations is that they preserve (a) point coincidence (e.g. line intersections), (b) parallelism between straight lines, and (c) proportions of distances

between points on parallel lines [26]. For example, consider the configuration of points p_a, p_b, p_c, p_d and lines l_1, l_2, l_3 in Fig. 5: (a) we cannot introduce new points of coincidence between lines by applying transformations such as translation, scaling, reflection, and rotation. Conversely, if two lines intersect, then they will still intersect after these transformations; (b) lines l_1, l_2 are parallel before and after the transformations; lines l_1, l_3 are always non-parallel; (c) the ratio of distances between collinear points p_a, p_b, p_c is maintained; formally, let s_{ij} be the segment between points p_i, p_j and let $|s_{ij}|$ be the length of the segment. Then the ratio $\frac{|s_{ab}|}{|s_{bc}|}$ in Fig. 5 is the same before and after the transformations.

Based on these properties we can determine the transformations that preserve various qualitative spatial relations.[7]

Theorem 1. *The following qualitative spatial relations are preserved under translation, scale, rotation, and reflection (applied to the embedding space): topology, mereology, coincidence, collinearity, line parallelism.*

Proof. By definition, affine transformations preserve *parallelism* with respect to qualitative line orientation, and point *coincidence*. Due to preservation of point coincidence and proportions of collinear distances by affine transformations, it follows that mereological *part of* and topological *contact* relations between regions are preserved, i.e. if a mereological or topological relation changes between regions a, b, then by definition there exists a point p coincident with a such that the coincidence relation between p and b has changed; but this cannot occur as point coincidence is maintained with affine transformations by definition, therefore mereological and topological relations are also maintained.

The interaction between spatial relations and transformations is richer than we have space to elaborate on here, i.e. not all qualitative spatial relations are preserved under all affine transformations; orientation is not preserved under reflection (e.g. Fig. 5(b) gives a counter example), distance is not preserved under non-uniform scaling. To summarise, we formalise the following knowledge as modular commonsense rules in CLP(QS): point-coincidence, line parallelism, topological and mereological relations are preserved with all affine transformations. Relative orientation changes with reflection, and qualitative distances and perpendicularity change with non-uniform scaling. Spheres, circles, and rectangles are not preserved with non-uniform scaling, with the exception of axis-aligned bounding boxes.

"Trading" Transformations. Symmetries are used to eliminate object variables. As a metaphor, unbound variables are replaced by constant values in

[7] The properties of affine transformations and the geometric objects that they preserve are well understood; further information is readily available in introductory texts such as [26]. Our key contribution is formalising and exploiting this spatial knowledge as modular and extensible common-sense rules in intelligent knowledge-based spatial assistance systems.

"exchange" for transformations. We start with a set of transformations that can be applied to a configuration: translation, scaling, arbitrary rotation, and horizontal and vertical reflection. We can then "trade" each transformation for an elimination of variables. Each transformation can only be "spent" once. Theorem 2 presents an instance of such a pruning case.

Table 2 presents a variety of different pruning cases for position variables and the associated combination of transformations, as illustrated in Fig. 6.[8] Some cases require more than one distinct set of parameter restrictions to cover the set of all position variables due to point coincidence being preserved by affine transformations. For example, consider case (f): all pairs of points p_1, p_2 can be transformed into any other pair of points p_i, p_j by translation, rotation, and scaling, iff $p_1 = p_2 \leftrightarrow p_i = p_j$. Thus, to cover all pairs of possible points, we need to consider two distinct parameter restrictions: $p_i = p_j$ and $p_i \neq p_j$; we refer to these as *subcases*.

Many further pruning cases are identifiable. For example, a version of case (i) can be defined without reflection by requiring more sub-cases where $c_4 > c_2$ and $c_4 < c_2$. Case (i) can be extended so that all six coordinates of three points are grounded if we also "exchange" the *skew* transformation (e.g. applicable to object domains like triangles or points).

Theorem 2. *Any pair of object position variables* $(x_1, y_1), (x_2, y_2)$ *can be transformed into any given position constants* $(c_1, c_2), (c_3, c_2)$ *such that* $(c_1 = c_3 \leftrightarrow (x_1 = x_2 \wedge y1 = y2))$ *by applying: an xy-translation, a rotation about the origin in the range* $(0, 2\pi)$*, and an x-scale.*

Proof. The corresponding expression has been verified using the *Reduce* system (*Redlog* quantifier elimination) [15]; all variables are quantified over reals.

$$
\begin{aligned}
&\forall c_1 \forall c_2 \forall c_3 \forall x_1 \forall y_1 \forall x_2 \forall y_2 \\
&(c_1 = c_3 \leftrightarrow (x_1 = x_2 \wedge y1 = y2)) \leftrightarrow \exists t_x \exists t_y \exists d_x \exists d_y \exists s_x \big(\\
&(0 < s_x) \wedge (d_x^2 + d_y^2 = 1) \wedge \\
&\text{let} S = \begin{pmatrix} s_x & 0 \\ 0 & 1 \end{pmatrix} \wedge \text{let } R = \begin{pmatrix} d_x - d_y \\ d_y \quad d_x \end{pmatrix} \wedge \text{let} T = \begin{pmatrix} t_x \\ t_y \end{pmatrix} \wedge \\
&\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = SR \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + T \wedge \begin{pmatrix} c_3 \\ c_2 \end{pmatrix} = SR \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} + T) \equiv \top
\end{aligned}
$$

We can use this pruning case on any spatial constraint graph G where the graph's spatial relations are preserved by translation, rotation, and scaling. Given graph $G = (N, E)$, the following Algorithm applies the pruning case, with selected constants $c_1 = 0$, $c_2 = 0$, and $c_3 = 1$ or $c_3 = 0$:

1. select object position variables p_1, p_2 from nodes in N
2. copy G to create G_1, G_2
3. in G_1 set $p_1 = (0, 0), p_2 = (1, 0)$ *(case $c_1 \neq c_3$)*
4. in G_2 set $p_1 = (0, 0), p_2 = (0, 0)$ *(case $c_1 = c_3$)*
5. if the task is:

[8] All cases have been verified using Reduce as presented in Theorem 2.

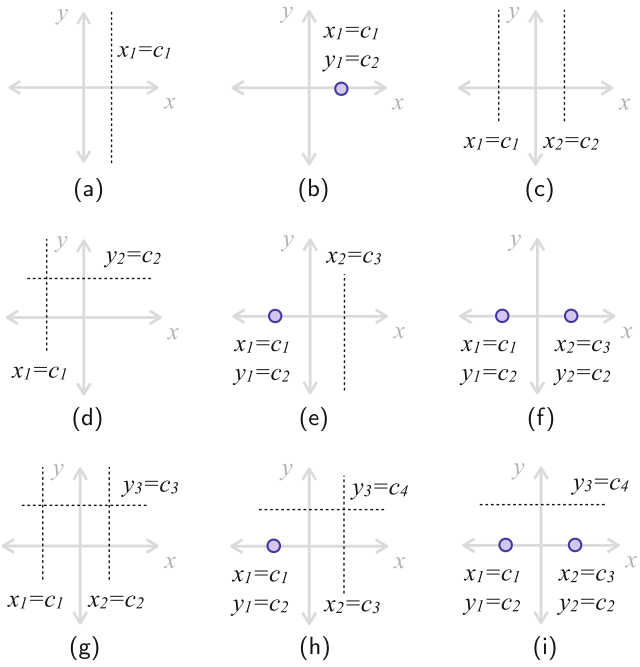


Fig. 6. Cases for pruning position parameters.

(a) *consistency* of G then solve $\bigvee_{i=1}^{2} \exists s\ G_i(s)$
(b) *sufficiency*, $G \rightarrow G'$, then solve
$\bigwedge_{i=1}^{2} \neg\exists s(G_i(s) \wedge \neg G'(s))$

In Step 1 any pair of objects can be selected for which their position variables will be grounded; we also employ policies that target computationally costly subgraphs (for example, pairs of non-equal circles that share a boundary point are often good candidates for this pruning case). Having eliminated free variables from the system of polynomial constraints, the constraints are significantly more simple to solve. Due to the double exponential complexity $O(c_1^{c_2^n})$ reducing n has a significant impact on performance; the system may even collapse from nonlinear constraints to linear (solvable in $O(c^n)$) or constants.

3.5 Combining Symmetry Pruning with Graph Decomposition

In certain cases, spatial constraint graphs can be decomposed into subgraphs that can be solved independently. For example, subgraphs G_1, G_2 can be independently solved if all objects in subgraph G_1 are either:

- *disconnected* from all objects in subgraph G_2;
- a *proper part* of some object in G_2;
- *left of* some segment in G_2;
- only related by *relative size* to some object in G_2, and so on.

In such cases we can reapply spatial symmetry pruning in each independent sub-graph; this commonsense spatial knowledge is modularly formalised within CLP(QS). For example, consider Proposition 22 of Book I of Euclid's Elements (Fig. 7):

Constructing a triangle from three segments. Given three line segments l_{ab}, l_{cd}, l_{ef}, draw a line through four collinear points $p_1, \ldots, p_4$ such that $|(p_1, p_2)| = |l_{ab}|$, $|(p_2, p_3)| = |l_{cd}|$, $|(p_3, p_4)| = |l_{ef}|$. Draw circle c_a centred on p_2, coincident with p_1. Draw circle c_b centred on p_3 coincident with p_4. Draw p_5 coincident with c_a and c_b. The triangle p_2, p_3, p_5 has side lengths such that $|(p_2, p_3)| = |l_{cd}|$, $|(p_3, p_5)| = |l_{ef}|$, $|(p_5, p_2)| = |l_{ab}|$.

In this example, the three segments l_{ab}, l_{cd}, l_{ef} and the remaining objects are only related by the distances between their end points. That is, the relative position and orientation of l_{ab}, l_{cd}, l_{ef} is not relevant to the consistency of the spatial graph; we only need to explore all combinations of segment *lengths*. Thus the solver decomposes the graph into four sub-graphs: (1) l_{ab} (2) l_{cd} (3) l_{ef}, and (4) $p_1, \ldots, p_5, c_a, c_b$. In subgraphs (1),(2),(3) it "trades" translation and rotation to ground $p_a = p_c = p_e = (0, 0)$, and $y_b = y_d = y_f = 0$ and keeps x-scale to cover all possible combinations of segment lengths, i.e. x_b, x_d, x_f are free variables. In subgraph (4) CLP(QS) applies the pruning case of Theorem 2 by grounding p_1, p_4.

Table 2. Cases for pruning parameters for one position point (a,b), two position points (c-f), three position points (g-i). Cases marked with $*$ require arbitrary scaling (i.e. both uniform and non-uniform).

Case	Parameter restrictions	Traded transformations
a	$x_1 = c_1$	x-translate
b	$x_1 = c_1, y_1 = c_2$	xy-translate
c	$x_1 = c_1, x_2 = c_2$, (i) $c_1 \neq c_2$(ii) $c_1 = c_2$	x-translate, rotate π, x-scale
d	$x_1 = c_1, y_2 = c_2$	xy-translate
e	$x_1 = c_1, y_1 = c_2, x_2 = c_3$, (i) $c_1 \neq c_3$(ii) $c_1 = c_3$	xy-translate, rotate π, x-scale
f	$x_1 = c_1, y_1 = c_2, x_2 = c_3, y_2 = c_2$, (i) $c_1 \neq c_3$(ii) $c_1 = c_3$	xy-translate, rotate $(0, 2\pi)$, x-scale
g	$x_1 = c_1, x_2 = c_2, y_3 = c_3$ (i) $c_1 \neq c_2$(ii) $c_1 = c_2$	xy-translate, rotate π, x-scale
h $*$	$x_1 = c_1, y_1 = c_2, x_2 = c_3, y_3 = c_4$ (i) $c_1 \neq c_3 \wedge c_2 \neq c_4$(ii) $c_1 = c_3 \wedge c2 \neq c_4$ (iii) $c_1 \neq c_3 \wedge c_2 = c_4$(iv) $c_1 = c_3 \wedge c2 = c_4$	xy-translate, rotate π, xy-scale, y-reflect
i $*$	$x_1 = c_1, y_1 = c_2, x_2 = c_3, y_2 = c_2, y_3 = c_4$ (i) $c_1 \neq c_3 \wedge c_2 \neq c_4$(ii) $c_1 \neq c_3 \wedge c_2 = c_4$ (iii) $c_1 = c_3 \wedge c_2 = c_4$	xy-translate, rotate $(0, 2\pi)$, xy-scale, y-reflect

```
?- Sab = segment(Pa,Pb), Scd = segment(Pc,Pd), Sef = segment(Pe,Pf),
|  S12 = segment(P1,P2), S23 = segment(P2,P3), S34 = segment(P3,P4),
|  S14 = segment(P1,P4),
|  coincident(P2,S14), coincident(P3,S14),
|
|  length_dim(equal, S12, Sab),
|  length_dim(equal, S23, Scd),
|  length_dim(equal, S34, Sef),
|
|  Ca=circle(P2,_),   coincident(P1,Ca),
|  Cb=circle(P3,_),   coincident(P4,Cb),
|  coincident(P5,Ca), coincident(P5,Cb),
|  S35 = segment(P3,P5), S52 = segment(P5,P2),
|
|  %% triangle
|  length_dim(equal, S23, Scd),
|  length_dim(equal, S35, Sef),
|  length_dim(equal, S52, Sab).
true.
...
|  (length_dim(not_equal, S23, Scd);
|   length_dim(not_equal, S35, Sef);
|   length_dim(not_equal, S52, Sab)).
false.
```

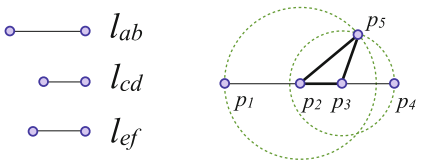


Fig. 7. Constructing a triangle by decomposing the spatial constraint graph.

4 Application-Driven Use Cases

We present problem instances in the classes of *mereology*, *ruler and compass*, and *contact*. Table 3 presents the experiment time results of CLP(QS) using symmetry pruning compared with existing systems: *z3* SMT solver, *Redlog* real quantifier elimination (in the *Reduce* computer algebra system) [15], and the relation algebraic qualitative spatial reasoners GQR [16] and SparQ [39]. CLP(QS) uses *z3* to solve polynomial constraints (after our pruning), thus z3 is the most direct comparison. Experiments were run on a MacBookPro, OS X 10.8.5, 2.6 GHz Intel Core i7. The empirical results show that no other spatial reasoning system exists (to the best of our knowledge) that can solve the range of problems presented in this section, and in cases where solvers are applicable, CLP(QS) with spatial pruning solves those problems significantly faster than other systems.

4.1 Spatial Theorem Proving: Geometry of Solids

Tarski [36] shows that a geometric point can be defined by a language of mereological relations over spheres. The idea is to distinguish when spheres are concentric, and to define a geometric point as the point of convergence. Borgo [8] shows

Table 3. Time (in seconds) to solve benchmark problems using CLP(QS) with pruning compared to z3 SMT solver, Redlog (Reduce) quantifier elimination, and GQR and SparQ relation algebraic solvers. *Time out* was issued after a running time of 10 min. Failure (*fail*) indicates that the incorrect result was given. Not applicable (*n/a*) indicates that the problem could not be expressed using the given system.

Problem	CLP(QS)	z3	Redlog	GQR	SparQ
Aligned Concentric	6.831	47.651	*time out*	*n/a*	*n/a*
Boundary Concentric	2.036	*time out*	*time out*	*n/a*	*n/a*
Mereologically Concentric	0.105	0.373	*time out*	*n/a*	*n/a*
Angle Bisector	0.931	*time out*	*time out*	*n/a*	*n/a*
Sphere Contact	0.004	*time out*	*time out*	*fail*	*fail*

that this can be accomplished with a language of mereology over *hypercubes*. We will use CLP(QS) to prove that the definitions are sound for rotatable squares.

As a preliminary we need to determine whether the intersection of two squares is non-square (Fig. 8) [34]. Given two squares a, b, the intersection is non-square if a *partially overlaps* b (Table 1) and either (a) a and b are not aligned, $x_{v_a} \neq x_{v_b}$ or (b) the width and height of the intersection are not equal, $w_I \neq h_I$, such that

$$w_I = \min(v \cdot p_{2_a}, v \cdot p_{2_b}) - \max(v \cdot p_{1_a}, v \cdot p_{1_b})$$

$$h_I = \min(v' \cdot p_{4_a}, v' \cdot p_{4_b}) - \max(v' \cdot p_{1_a}, v' \cdot p_{1_b})$$

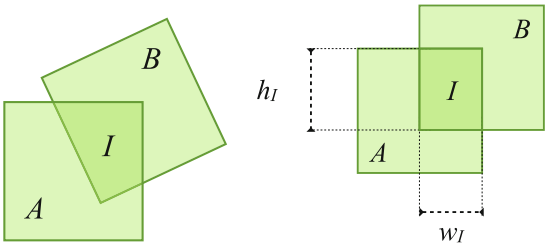
Aligned Concentric. Two squares A, B are aligned and concentric if: A is *part of* B and there does not exist a square P such that (a) P is covertex with B, and (b) the intersection of P and A is not a square (Fig. 9).

```
|  aligned_concentric(A,B) :-
|   mereology(part_of, A,B),
|   not((
|    P=square(_,_,_),
|    covertex(P,B),
|    intersection(non_square,A,P).
|   )).
```

We use CLP(QS) to prove that the definition is sufficient, by contradiction; two squares are covertex if they are aligned and share a vertex, and the relation *concentric* is the geometric definition of concentricity in Table 1 that is used to evaluate the mereological definition of concentricity:

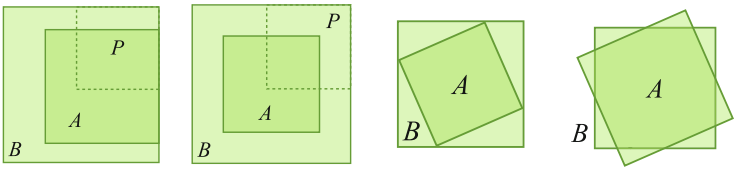
```
?- A=square(_,_,_),
|  B=square(_,_,_),
|  aligned(A,B),
|  not_concentric(A,B),
|  aligned_concentric(A,B).
false.
```

Boundary Concentric. Square A is boundary concentric with square B if: A is *proper part of* B and there does not exist a square Z such that (a) Z is *proper part of* B (b) A is *part of* Z, and (c) Z is not *part of* A (Fig. 9).



(a) A, B are not aligned. (b) $h_I \neq w_I$

Fig. 8. Intersection I of squares A, B is non-square.



(a) A and B are non-concentric (b) A and B are concentric (c) A is boundary concentric with B. (d) A and B are rotated concentric.

Fig. 9. Characterising aligned (a), (b) and rotated (c), (d) concentric squares using mereology (reproduced from [Borgo, 2013])

```
|  boundary_concentric(A,B) :-
|   mereology(proper_part,A,B),
|   not((
|    Z = square(_,_,_)
|    mereology(proper_part, Z,B),
|    mereology(part_of, A,Z),
|    mereology(not_part_of, Z,A),
|   )).
```

```
?- A=square(_,_,_),
|  B=square(_,_,_),
|  not_aligned(A,B),
|  not_concentric(A,B),
|  boundary_concentric(A,B).
false.
```

Mereologically Concentric. Squares A, B are mereologically concentric if: A, B are *aligned concentric* or there exists Q such that (a) Q is *boundary concentric* with B and Q is *aligned concentric* with A or (b) Q is *boundary concentric* with A and Q is *aligned concentric* with B.

Having proved the mereological definitions of *aligned* and *boundary* concentricity, we can replace these with more efficient geometric definitions from Table 1 when proving mereological concentricity.

```
|  g_aligned_concentric(A,B) :-
|   aligned(A,B), concentric(A,B).
|
|  g_boundary_concentric(A,B) :-
|   mereology(boundary_part, A,B).
```

```
|  m_concentric(A,B) :-
|   g_aligned_concentric(A,B)
|   ;
|   Q = square(_,_,_),
|   g_boundary_concentric(Q,B),
|   g_aligned_concentric(Q,A)
|   ;
|   Q = square(_,_,_),
|   g_boundary_concentric(Q,A),
|   g_aligned_concentric(Q,B).
```

```
?- A=square(_,_,_),
|  B=square(_,_,_),
|  not_concentric(A,B),
|  m_concentric(A,B).
false.
```

4.2 Didactics: Ruler–Compass and Contact Problems

Angle Bisector. Let l_a, l_b be line segments that share an endpoint at p. Draw circle c at p. Circle c intersects l_a at p_a and l_b at p_b. Draw circles c_a at p_a and circle c_b at p_b such that p is coincident with both c_a, c_b. Circles c_a and c_b intersect at p and p_c. The line segment from p to p_c bisects the angle between l_a and l_b (Fig. 10).

We use CLP(QS) to prove that the definition is sufficient. The relation *bisects* is used for evaluation by checking if the midpoint of (p_a,p_b) is collinear with l_c (i.e. idealised rulers cannot directly measure the midpoint of a line). An interactive diagram is then automatically generated that encodes the specified program (using the FreeCAD system); see Fig. 11.

```
?- La = segment(P,_),Lb = segment(P,_),
|  orientation(not_parallel, La, Lb),
|  C = circle(P,_),
|  coincident(Pa,La),coincident(Pa,C),
|  coincident(Pb,Lb),coincident(Pb,C),
|  Ca=circle(Pa,_), Cb=circle(Pb,_),
|  coincident(Pa,Ca),coincident(Pa,Cb),
|  coincident(Pc,Ca),coincident(Pc,Cb),
|  not_equal(P,Pc),
|  Lc=segment(P,Pc),
|  bisects(segment(Pa,Pb),Lc).
true.
...
|    not_bisects(segment(Pa,Pb),Lc).
false.
```

Sphere Contact. Determine the maximum number of same-sized mutually touching spheres (Fig. 4 - note that no numeric information about the spheres is given in this benchmark problem).

```
?- sphere_list(5, Spheres),
|  size(equal, group(Spheres),
|  topology(rcc8(ec), group(Spheres) ).
false.
```

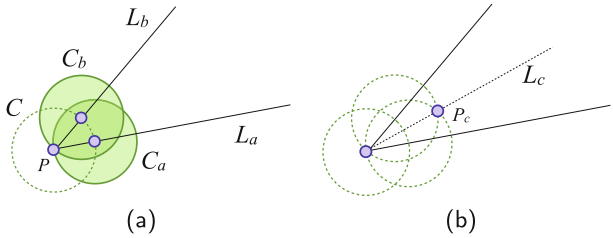


Fig. 10. Ruler and compass method for angle bisection. Line L_c bisects the angle between lines L_a, L_b.

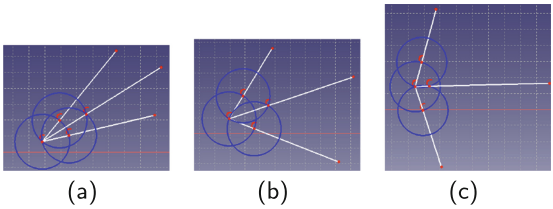


Fig. 11. Interactive diagram encoding the student's constructive proof of Euclid's angle bisector theorem; as the student manipulates figures in the diagram, the other geometries are automatically updated to maintain the specified qualitative constraints.

5 Conclusions

Affine transformations provide an effective and interesting class of symmetries that can be used for pruning across a range of qualitative spatial relations. To summarise, we formalise the following knowledge as modular commonsense rules in CLP(QS): point-coincidence, line parallelism, topological and mereological relations are preserved with all affine transformations. Relative orientation changes with reflection, and qualitative distances and perpendicularity change with non-uniform scaling. Spheres, circles, and rectangles are not preserved with non-uniform scaling, with the exception of axis-aligned bounding boxes. Our algorithm is simple to implement, and is easily extended to handle more pruning cases.

Theoretical and empirical results show that our method of pruning yields an improvement in performance by orders of magnitude over standard polynomial encodings without loss of soundness, thus increasing the horizon of spatial problems solvable with *any* polynomial constraint solver. Furthermore, the declaratively formalised knowledge about pruning strategies is available to be utilised in a modular manner within other knowledge representation and reasoning frameworks that rely on specialised SMT solvers etc., e.g., in the manner demonstrated in ASPMT(QS) [38], which is a specialised non-monotonic spatial reasoning system built on top of answer set programming modulo theories.

References

1. Aiello, M., Pratt-Hartmann, I.E., van Benthem, J.F.: Handbook of Spatial Logics. Springer-Verlag New York Inc., Secaucus (2007). ISBN 978-1-4020-5586-7
2. Arnon, D.S., Collins, G.E., McCallum, S.: Cylindrical algebraic decomposition I: the basic algorithm. SIAM J. Comput. **13**(4), 865–877 (1984)
3. Bhatt, M., Wallgrün, J.O.: Geospatial narratives and their spatio-temporal dynamics: Commonsense reasoning for high-level analyses in geographic information systems. ISPRS Int. J. Geo-Information **3**(1), 166–205 (2014). doi:10.3390/ijgi3010166. http://dx.doi.org/10.3390/ijgi3010166
4. Bhatt, M., Lee, J.H., Schultz, C.: CLP(QS): a declarative spatial reasoning framework. In: Egenhofer, M., Giudice, N., Moratz, R., Worboys, M. (eds.) COSIT 2011. LNCS, vol. 6899, pp. 210–230. Springer, Heidelberg (2011)
5. Bhatt, M., Schultz, C., Freksa, C.: The 'Space' in spatial assistance systems: conception, formalisation and computation. In: Tenbrink, T., Wiener, J., Claramunt, C. (eds.) Representing space in cognition: Interrelations of behavior, language, and formal models. Series: Explorations in Language and Space. Oxford University Press (2013). 978-0-19-967991-1
6. Bhatt, M., Suchan, J., Schultz, C.: Cognitive interpretation of everyday activities - toward perceptual narrative based visuo-spatial scene interpretation. In: Finlayson, M., Fisseni, B., Loewe, B., Meister, J.C. (eds.) Computational Models of Narrative (CMN) 2013, a satellite workshop of CogSci 2013: The 35th meeting of the Cognitive Science Society., Dagstuhl, Germany, OpenAccess Series in Informatics (OASIcs) (2013)
7. Bhatt, M., Schultz, C.P.L., Thosar, M.: Computing narratives of cognitive user experience for building design analysis: KR for industry scale computer-aided architecture design. In: Baral, C., Giacomo, G.D., Eiter, T. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, 20–24 July, 2014. AAAI Press (2014). ISBN 978-1-57735-657-8
8. Borgo, S.: Spheres, cubes and simplexes in mereogeometry. Logic Logical Philos. **22**(3), 255–293 (2013)
9. Bouhineau, D.: Solving geometrical constraint systems using CLP based on linear constraint solver. In: Pfalzgraf, J., Calmet, J., Campbell, J. (eds.) AISMC 1996. LNCS, vol. 1138, pp. 274–288. Springer, Heidelberg (1996)
10. Bouhineau, D., Trilling, L., Cohen, J.: An application of CLP: Checking the correctness of theorems in geometry. Constraints **4**(4), 383–405 (1999)
11. Buchberger, B.: Bruno Buchberger's PhD thesis 1965: an algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal (English translation). J. Symbolic Comput. **41**(3), 475–511 (2006)
12. Chou, S.-C.: Mechanical Geometry Theorem Proving, vol. 41. Springer Science and Business Media, Dordrecht (1988)
13. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In: Brakhage, H. (ed.) GI-Fachtagung 1975. LNCS, vol. 33, pp. 134–183. Springer, Heidelberg (1975)
14. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. J. Symbolic Comput. **12**(3), 299–328 (1991). ISSN 0747–7171
15. Dolzmann, A., Seidl, A., Sturm, T.: REDLOG User Manual, Edition 3.0, Apr 2004
16. Gantner, Z., Westphal, M., Wölfl, S.: GQR-A fast reasoner for binary qualitative constraint calculi. In: Proceedings of AAAI, vol. 8 (2008)

17. Hadas, N., Hershkowitz, R., Schwarz, B.B.: The role of contradiction and uncertainty in promoting the need to prove in dynamic geometry environments. Educ. Stud. Mathe. **44**(1–2), 127–150 (2000)
18. Haunold, P., Grumbach, S., Kuper, G., Lacroix, Z.: Linear constraints: Geometric objects represented by inequalitiesl. In: Frank, A.U. (ed.) COSIT 1997. LNCS, vol. 1329, pp. 429–440. Springer, Heidelberg (1997)
19. Jaffar, J., Maher, M.J.: Constraint logic programming: A survey. J. Logic Prog. **19**, 503–581 (1994)
20. Kanellakis, P.C., Kuper, G.M., Revesz, P.Z.: Constraint query languages. In: Rosenkrantz, D.J., Sagiv, Y. (eds.) Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Nashville, Tennessee, USA, 2–4 April, 1990, pp. 299–313. ACM Press (1990). ISBN 0-89791-352-3
21. Kapur, D., Mundy, J.L. (eds.): Geometric Reasoning. MIT Press, Cambridge (1988). ISBN 0-262-61058-2
22. Ladkin, P.B., Maddux, R.D.: On binary constraint problems. J. ACM (JACM) **41**(3), 435–469 (1994)
23. Lee, J.H.: The complexity of reasoning with relative directions. In: 21st European Conference on Artificial Intelligence (ECAI 2014) (2014)
24. Ligozat, G.: Qualitative Spatial and Temporal Reasoning. Wiley-ISTE, Hoboken (2011)
25. Ligozat, G.F.: Qualitative triangulation for spatial reasoning. In: Campari, I., Frank, A.U. (eds.) COSIT 1993. LNCS, vol. 716, pp. 54–68. Springer, Heidelberg (1993)
26. Martin, G.E.: Transformation geometry: An introduction to symmetry. Springer, New York (1982)
27. Owen, J.C.: Algebraic solution for geometry from dimensional constraints. In: Proceedings of the First ACM Symposium on Solid Modeling Foundations and CAD/CAM Applications, pp. 397–407. ACM (1991)
28. Pesant, G., Boyer, M.: QUAD-CLP (R): Adding the power of quadratic constraints. In: Borning, A. (ed.) PPCP 1994. LNCS, vol. 874, pp. 95–108. Springer, Heidelberg (1994)
29. Pesant, G., Boyer, M.: Reasoning about solids using constraint logic programming. J. Automated Reasoning **22**(3), 241–262 (1999)
30. Randell, D.A., Cohn, A.G., Cui Z.: Computing transitivity tables: A challenge for automated theorem provers. In 11th International Conference on Automated Deduction (CADE-11), pp. 786–790 (1992)
31. Ratschan, S.: Approximate quantified constraint solving by cylindrical box decomposition. Reliable Comput. **8**(1), 21–42 (2002)
32. Ratschan, S.: Efficient solving of quantified inequality constraints over the real numbers. ACM Trans. Comput. Logic (TOCL) **7**(4), 723–748 (2006)
33. Schultz, C., Bhatt, M.: Towards a declarative spatial reasoning system. In: 20th European Conference on Artificial Intelligence (ECAI 2012) (2012)
34. Schultz, C., Bhatt, M.: Declarative spatial reasoning with boolean combinations of axis-aligned rectangular polytopes. In: ECAI 2014–21st European Conference on Artificial Intelligence, pp. 795–800 (2014)
35. Schultz, C., Bhatt, M., Borrmann, A.: Bridging qualitative spatial constraints and parametric design - a use case with visibility constraints. In: EG-ICE: 21st International Workshop - Intelligent Computing in Engineering 2014 (2014)
36. Tarski, A.: A general theorem concerning primitive notions of Euclidean geometry. Indagationes Mathematicae **18**(468), 74 (1956)

37. Varzi, A.C.: Parts, wholes, and part-whole relations: The prospects of mereotopology. Data Knowl. Eng. **20**(3), 259–286 (1996)
38. Walega, P., Bhatt, M., Schultz, C.: ASPMT(QS): non-monotonic spatial reasoning with answer set programming modulo theories. In: LPNMR: Logic Programming and Nonmonotonic Reasoning - 13th International Conference (2015). http://lpnmr2015.mat.unical.it
39. Wallgrün, J.O., Frommberger, L., Wolter, D., Dylla, F., Freksa, C.: Qualitative spatial representation and reasoning in the SparQ-Toolbox. In: Barkowsky, T., Knauff, M., Ligozat, G., Montello, D.R. (eds.) Spatial Cognition 2007. LNCS (LNAI), vol. 4387, pp. 39–58. Springer, Heidelberg (2007)
40. Wenjun, W.: Basic principles of mechanical theorem proving in elementary geometries. J. Syst. Sci. Math. Sci. **4**(3), 207–235 (1984)