

Maintaining Spatial Relations in an Incremental Diagrammatic Reasoner

Ronald W. Ferguson, Joseph L. Bokor, Rudolph L. Mappus IV, and Adam Feldman

College of Computing, Georgia Institute of Technology
Atlanta, GA 30332 USA

{rwf, jlbokor, cmappus, storm}@cc.gatech.edu

Abstract. This paper describes an architecture for dynamically handling spatial relations in an incremental, nonmonotonic diagrammatic reasoning system. The architecture represents jointly exhaustive and pairwise disjoint (JEPD) spatial relation sets as nodes in a dependency network. These spatial relation sets include interval relations, relative orientation relations, and connectivity relations, but in theory could include any JEPD spatial relation sets. This network then caches dependencies between low-level spatial relations, allowing those relations to be easily assumed or retracted as visual elements are added or removed from a diagram. For example, in the architecture's Undo mechanism, the dependency network can quickly reactivate cached spatial relations when a previously-deleted element is restored. As part of this work, we describe how the system supports higher-level reasoning, including support for creating default assumptions. We also describe how this system was integrated with an existing drawing program and discuss its possible use in diagrammatic and geographic reasoning.

1 Introduction

Diagrams are useful in a wide variety of reasoning tasks. Because a single diagram can convey many spatial relations at a glance, diagrams provide rich sources of information for a number of domains, including geographic, architectural, and engineering domains.

This capability is more interesting when we consider that the spatial relations in a diagram need not be static. Diagrams frequently change over time. The additions, removals, and modifications of elements also change the set of spatial relations. Handling such incremental changes without significant reprocessing of previously established spatial relations is key to making spatial and diagrammatic reasoning efficient. Incremental processing is also key to many geographic reasoning tasks. Maps indicating changing conditions, such as those for crisis planning, may be edited as conditions and needs change.

An incremental processing approach may also be driven by other factors. For example, problem solving with diagrams often requires many changes to a particular diagram over time as new ideas or subtasks emerge. The ability to understand the consequences of a change in a set of elements in conceptual terms may be affected by low-level changes. Finally, a more practical benefit of incremental processing is that it dis-

R. W. Ferguson, J. L. Bokor, R. L. Mappus, IV, and A. Feldman, "Maintaining spatial relations in an incremental diagrammatic reasoner," in *COSIT 2003, Spatial Information Theory*, W. Kuhn, M. Worboys, and S. Timpf, Eds.: Springer-Verlag, 2003.

tributes the processing burden more evenly over the extent of the task, which is useful on low-end devices, such as personal digital assistants.

In this paper, we describe an architecture for maintaining the set of qualitative spatial relations during incremental, nonmonotonic changes to a diagram. This work builds on earlier work on the GeoRep diagrammatic reasoner [1], which is described in the next section (after reviewing related research). After describing GeoRep, we discuss how GeoRep was modified to allow incremental processing, and cover a number of key theoretical and architectural issues: how to handle composite objects, the interface between low-level and high-level reasoning, and why such a system requires a new approach to handling default visual interpretations. We then discuss related work and future challenges for this system.

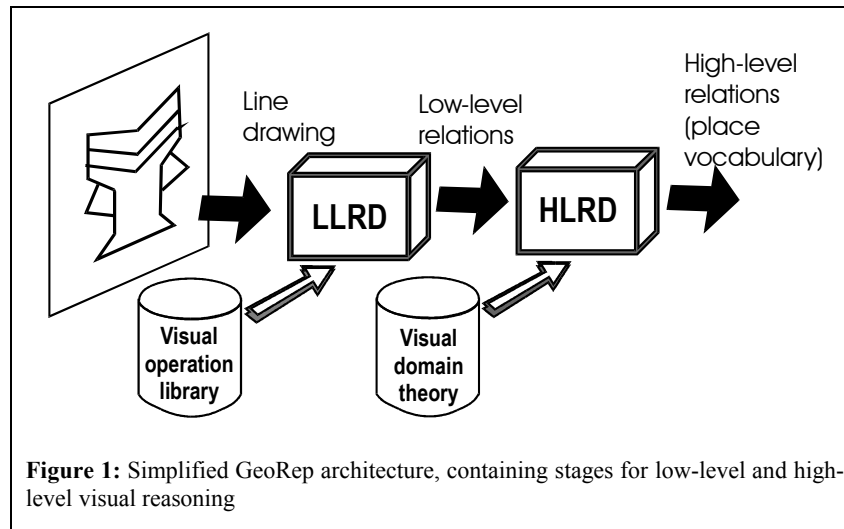
2 Related Work

A number of other researchers have looked at how incremental changes impact spatial reasoning.

In qualitative spatial reasoning, researchers have explored how to process qualitative spatial vocabularies incrementally. Notably, Hernández [2] proposed a mechanism for maintaining a consistent spatial description consisting of a combined set of orientational and topological spatial relations. Through careful analysis, separate orientation and topological relations sets are combined into a single vocabulary. These mechanisms also use a dependency network, similar to that in a truth-maintenance system, to allow relations to be added or deleted incrementally. At the same time, this mechanism does not address visual descriptions that contain multiple heterogeneous relation sets.

A number of researchers have explored the use of conceptual neighborhoods [3] and topological distances [4] to understand gradual change in the context of qualitative spatial vocabularies. Egenhofer and Al-Taha, for example, show how an analysis of topological distance between members of a relation set can be used to construct a graph that links the closest qualitative spatial relations. This graph can be used to show the set of necessary intervening qualitative states that must occur between two given states.

Along with this research in qualitative spatial reasoning, there are a number of sketching systems that must maintain knowledge of the links between individual visual elements and the inferred relations, although few of them have explicit frameworks for handling dependencies between spatial relations. A family of sketch interpretation systems by Davis and colleagues [5] use blackboard systems to integrate a low-level reasoner with a high-level description language, as with GeoRep, but can handle sketched shapes as well as vector graphics. Similarly, the Geometer's Sketchpad [6] uses a constraint network to enforce a set of constraints over a set of visual elements, which include line segments, rays, and circles. These constraints, however, are not discovered by the system, but must be entered by hand.



3 The GeoRep Diagrammatic Reasoner

This work extends an existing diagrammatic reasoner, GeoRep¹, to make processing incremental. GeoRep [1] has been used in a number of diagrammatic reasoning domains, including map-based military diagrams [7], simple physical diagrams [8], and logic circuits [9]. In addition, it has been used as the visual representation system for several cognitive modeling simulations [10, 11]. We first describe the GeoRep reasoner and then turn to the incremental processing mechanism.

GeoRep's input is a line-drawn figure, given as a vector graphics file. The vector graphics file contains a number of visual element types, including line segments, circles, ellipses, arcs, spline curves, and positioned text. As output, GeoRep produces a predicate calculus representation. This representation has three parts: the low-level spatial relations, the high-level interpretation of the figure, and intermediate spatial and conceptual relations that are produced in the process of interpretation.

To generate this description, GeoRep uses a two-stage architecture (Figure 1). We describe each stage in turn.

The first stage, the *Low-Level Relational Describer* (LLRD) represents a set of low-level spatial relations in the figure (Figure 2). These low-level relations are generated in a pipelined architecture, starting with simple proximity calculations and ending with more sophisticated relations, such as interval relations between parallel line segments.

These particular spatial relations are also designed to model those qualitative spatial relationships that are detected in early vision. For example, it is well-known that humans are sensitive to relative angles (such as perpendicular lines), indentations in fig-

¹ GeoRep is short for GEOMETRIC REPresenter, although it has also been used for geographic domains, such as reasoning about battlefield movements [7].

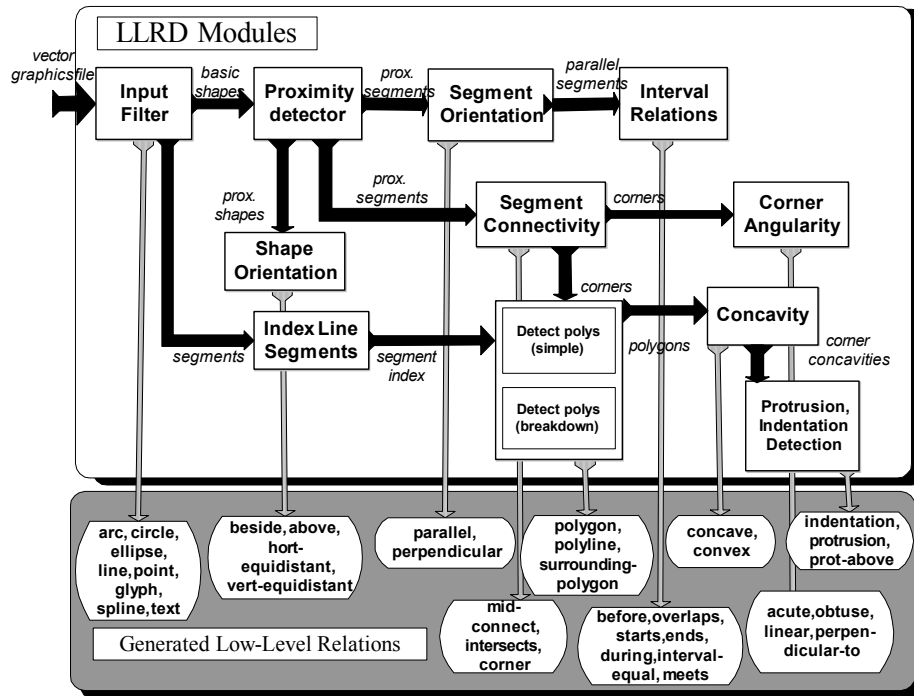
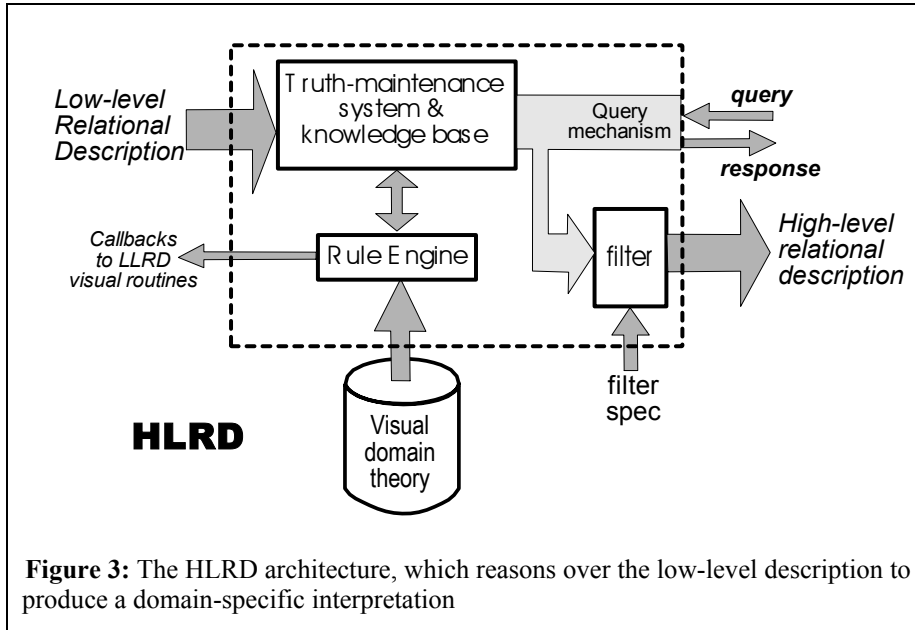


Figure 2: Data flow diagram of GeoRep’s Low-Level Relation Descriptor (LLRD). The top portion shows the collections of visual operators and the visual elements they process. The lower portion shows the set of spatial relations produced by each set of visual operators

ure boundaries [12], and to vertical and horizontal orientations in the assumed frame of reference [13]. Interestingly, one relation set used that has not been tested for early vision is interval relations [14] between parallel lines. In practice, these relations are extremely useful in modeling aspects of visual perception such as the detection of qualitative symmetry [15]. A simple attention model uses a proximity detector to limit visual relation tests to proximate visual elements. This acts as a limited focusing mechanism that keeps processing tractable.

The system also models some aspects of attention and perceptual organization, though in a domain-dependent fashion. “Grouping rules” can be used to simulate similarity-based grouping. Similarly, multiple LLRDs can be used to simulate visual separation based on factors such as color [9].

The second stage, the *High-Level Relational Descriptor* (HLRD; Figure 3), uses these low-level relations and a rule-based *visual domain theory* to produce a description of the diagram in question. The output of the HLRD is a description of the figure expressed in a domain-dependent high-level representation. For example, using domain-dependent rules, the HLRD produces the high-level representation of the logic circuit in Figure 4, describing the gates, the inputs and outputs, and the input and output



labels. It has also been used in map-based military diagrams, called Course of Action (COA) diagrams (Figure 5, with a portion of the generated representation in Figure 6).

The HLRD rules utilize a pattern-directed inference system that is supported by a *logic-based truth maintenance system* (LTMS) [16]. HLRD rules can match against low-level spatial relations produced by the LLRD, as well as domain knowledge about the diagram (such as a small ontology of logic gate or military unit types). HLRD rules are specialized for spatial domains. They typically are constrained to run only on

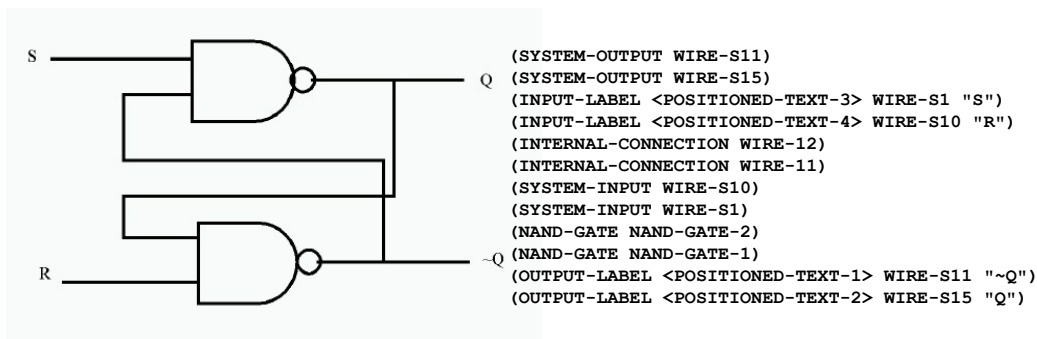


Figure 4: SR-Latch logic diagram and the representation produced by GeoRep.

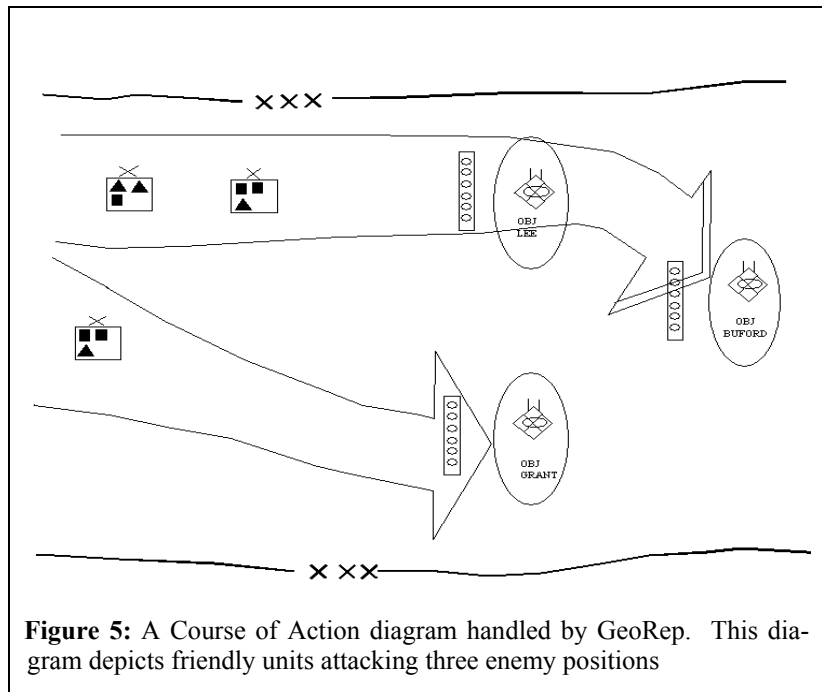


Figure 5: A Course of Action diagram handled by GeoRep. This diagram depicts friendly units attacking three enemy positions

proximate visual elements, and can call visual operations within the LLRD.

One limitation of GeoRep is its model of processing. GeoRep processes figures only in batch mode, and as a result, can run only on static diagrams. This is due to limitations of the LLRD rather than the HLRD. The HLRD is inherently incremental because it is based on an LTMS. The HLRD's relations can be assumed or retracted at any time, and the consequences of these relations will also be assumed or retracted accordingly. However, once the LLRD detects a visual relation between elements, it cannot retract it. While the use of visual tests is pipelined, the system does not store information

<p>Enemy unit inside area "Grant" (armor-unit unit-6) (enemy-unit unit-6) (military-unit unit-6) (motorized-rifle-unit unit-6) (unit-at unit-6 (objective-area area19 "Grant"))</p>	<p>Subordinate units for the task force (subordinate-maneuver-unit unit-10 unit-17) (mechanized-infantry-unit unit-10) (battalion unit-10) (friendly-unit unit-10) (subordinate-maneuver-unit unit-13 unit-17) (mechanized-infantry-unit unit-13) (battalion unit-13) (friendly-unit unit-13) (subordinate-maneuver-unit unit-15 unit-17) (armor-unit unit-15) (battalion unit-15) (friendly-unit unit-15)</p>	<p>Other (completed-minefield minefield320) Representation links <i>Bottom arrow</i> (represents (composite <polyline:4> (axis-of-advance-support attack4)) <i>Echelon marker</i> (represents (x-mark <segment:71> <segment:72>) (marker marker317 brigade)) <i>Mech unit inside unit-17</i> (represents <polygon:10> (mechanized-infantry-unit unit-10)) <i>Armor unit</i> (represents (composite <polygon:6> <ellipse:23>) (armor-unit unit-6))</p>
<p>Attack on area "Grant" (objective-area area19 "Grant") (area area19) (attack-on attack4 area19)</p>		
<p>Southernmost friendly task force (composite-unit unit-17) (brigade unit-17) (friendly-unit unit-17)</p>		

Figure 6: Sample subset of representations generated for Figure 5

about which visual elements are used in particular spatial relations.

Therefore, to make GeoRep incremental, it suffices to make the LLRD incremental. Our solution is to treat low-level elements and relations as assumptions that can be added or retracted as the diagram is modified by a user

4 Making GeoRep Incremental

Creating an incremental LLRD requires a number of modifications to the existing system. First, a dependency network for spatial relations is needed. Along with handling low-level relations, the network must also handle nonmonotonic inferences created by particular composite objects, such as polygons. The second part of making the LLRD incremental is to ensure that incremental changes generated by the LLRD are properly handled by the HLRD. This in turn requires a new mechanism for handling default object interpretations, which cannot be handled with the HLRD's existing truth-maintenance system.

4.1 Creating a Dependency Network within the LLRD

The dependency network we developed for the incremental LLRD draws on previous research on the mathematical character of qualitative spatial vocabularies that are jointly exhaustive and pairwise disjoint (JEPD) [17]. By ensuring the JEPD character of each node's relation set, this network can take advantage of a number of such vocabularies shown to be JEPD, such as interval relations [14] and RCC [18, 19]. The logical properties of these JEPD sets are important because they allow the dependency network to isolate a set of spatial relations relative to other possible relations.

To handle the incremental processing of input, we have re-designed the LLRD as a dependency network (Figure 7). This dependency network is responsible for several tasks. It tracks the low-level relations supported by each visual element, allows visual elements to be added or removed, and maintains the information needed to allow re-evaluation when elements are modified.

Each node in the network represents a set of alternative spatial relations – a single relation set that is JEPD. The node may be IN or OUT. If the node is IN, then one internal relation in the set is true. If the node is OUT, none of the internal relations are true. For example, each *relative-angle* node in Figure 7 must select one of three possible internal relations – *perpendicular*, *parallel*, or *skew* – to describe an angle relationship. Similarly, the *interval* node indicates that two line segments have one of seven interval relations [14].

The links between nodes allow antecedent relations to support consequent nodes. In this network, antecedent nodes represent how some spatial relations support the existence of one of a set of mutually exclusive alternatives.

Each internal relation in a node supports zero or more successor nodes. For example, a *parallel* internal relation supports the construction of an *interval* node. In other words, the internal relation combined with a visual test supports the whole truth value and labeling of the successor node.

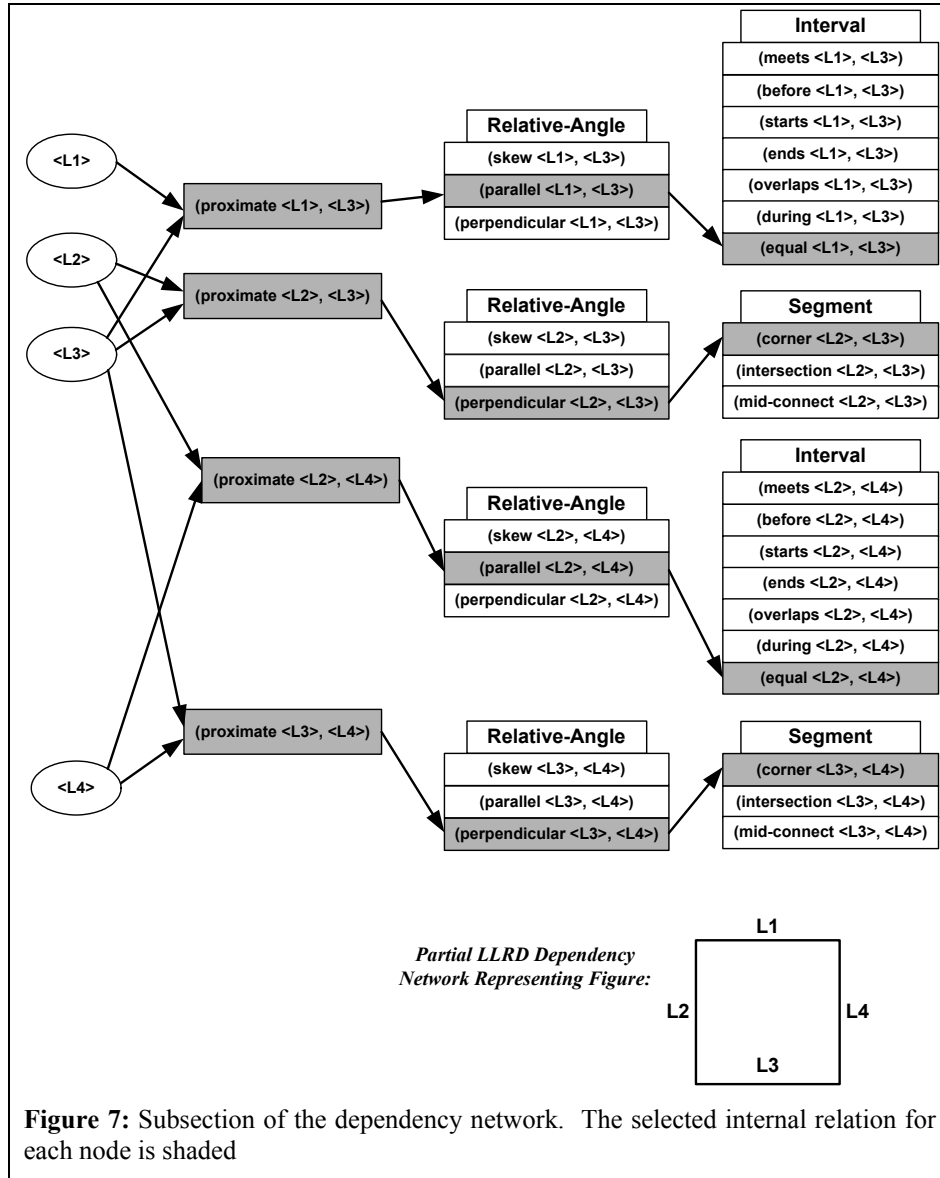


Figure 7: Subsection of the dependency network. The selected internal relation for each node is shaded

Let M be the set of nodes in the LLRD dependency network. Each node $m \in M$ has a truth value (IN or OUT). If m is IN, it also has an internal relation, which is one of n possible relations in a JEPD set.

It is important to note that the truth value of the network is not a relationship between the truth values of the nodes, but between the nodes, their antecedents, and the visual tests that are performed for each node's set of internal relations. For example,

for the *interval-equal* relation over segments L1 and L2, we can decompose the set of relations using the dependency network as follows:

$$\begin{aligned} \text{interval-equal}(L1,L2) &\equiv \text{parallel}(L1,L2) \wedge \text{interval-test}(L1,L2,\text{interval-equal}). \\ \text{parallel}(L1,L2) &\equiv \text{proximate}(L1,L2) \wedge \text{relative-angle-test}(L1,L2,\text{parallel}). \\ \text{proximate}(L1,L2) &\equiv L1 \wedge L2 \wedge \text{proximate-test}(L1,L2). \end{aligned}$$

Therefore:

$$\begin{aligned} \text{interval-equal}(L1,L2) &\leftrightarrow \\ &L1 \wedge L2 \wedge \text{proximate-test}(L1,L2) \wedge \text{relative-angle-test}(L1,L2,\text{parallel}) \\ &\wedge \text{interval-test}(L1,L2,\text{interval-equal}). \end{aligned}$$

This is equivalent to the set of tests applied to segments in the original pipelined version of the LLRD architecture.

Other types of relations are handled somewhat differently. Boolean relations are handled as JEPD sets with one element. *Proximate* is one relation handled in this fashion.

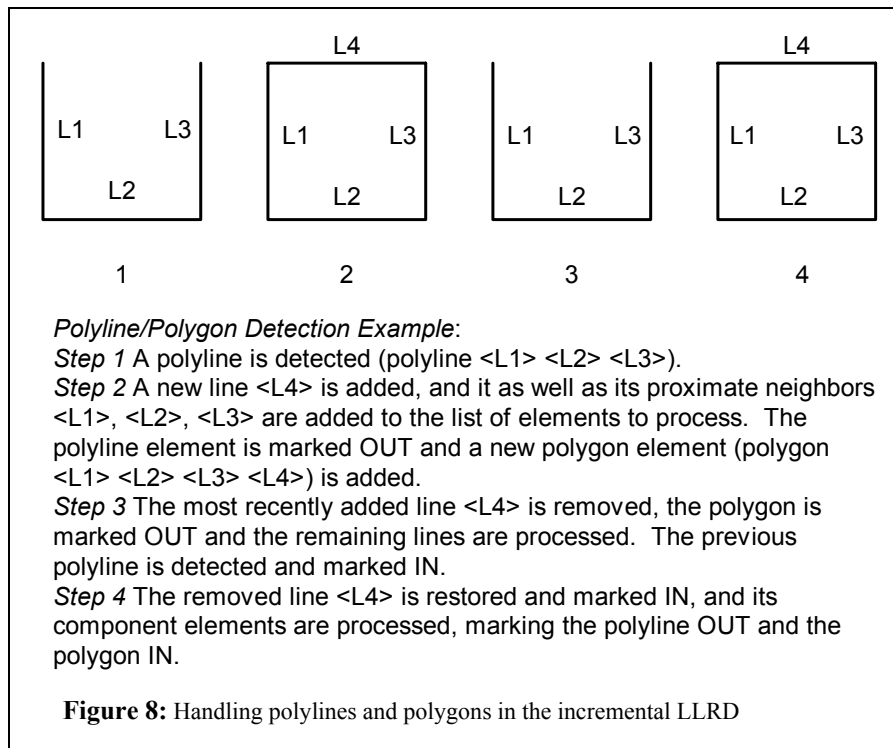
4.2 Handling Composite Objects

Composite objects (objects composed of multiple visual elements, such as polygons used to depict regions) are a special problem in incremental spatial reasoning [25].² Composite objects are shapes, but can also be treated as relations, since composite objects are detected based on their constituent elements. As a result, they are the only visual objects that can be retracted due to the retraction of other elements. In addition, the retraction of a visual element can lead to the detection or re-assumption of other composite objects. For example, removing a single line segment from a square will lead to its interpretation as a polyline. The LLRD currently handles two forms of composite objects: polylines and polygons.

For this reason, the incremental addition or removal of visual elements introduces new situations where polyline and polygon detection must be reapplied to existing visual elements. For example, when line segments are removed, the polylines and polygons that contained them must be reconsidered. Removing a line segment from a polyline or polygon can result in several possible combinations of polylines.

The LLRD performs polyline and polygon detection by using a *vertex index table*. As new line segments are added, the LLRD maintains the table of added segments indexed by endpoint. This table is then used to determine which line segments share endpoints with others. Groups of line segments that are not closed form polylines. Once a polyline is detected, it is added to the dependency network, and its vertices are re-

² It is, of course, possible to take an opposite strategy, treating composite elements as primary and non-decomposable. This avoids the problem with incrementality described here. GeoRep has a *glyph* element that works in this fashion, and which has been used to reason over figures with complex but self-contained shape descriptions [8]. However, composability is a hard problem to avoid entirely, and so we are using this simpler form of composability to delineate the challenges of more complex composability types.



moved from the table. Polygon detection uses the remaining entries in the table. If a set of vertices is closed, then a polygon is added to the dependency network and its vertices are removed from the table.

A node in the network for a composite object is not a relation node in our implementation. Instead, it represents the object by storing geometric information about the shape as well as linking the node to its constituent (subsumed) elements.

The LLRD uses different methods to determine which elements to reconsider for composite objects depending on whether an element is added, removed, or restored. In the first case, when a line segment is created, a new node is added and set to IN. Existing elements that are proximate to the added segment are added to the vertex index table in order to discover new polylines or polygons or changes to existing polylines. In contrast, when a line segment is retracted, the dependency network is used to retract any affected polylines and polygons. Lines that are part of the affected polyline or polygon, yet remain IN (i.e., have not been removed) are re-analyzed, and new polylines may be assumed. Finally, when an element is restored, the dependency network reactivates composite objects containing the restored element if all their subsumed elements are IN.

An example of how the LLRD handles composite objects is given in Figure 8. In step 1, line segments are added to form a polyline. In step 2, another segment is added, closing this polyline to form a new polygon. The polyline composite element is now OUT, and the new polygon is IN. In step 3, L4 is removed (OUT), and the polygon be-

comes OUT. The polyline formed by {L1 L2 L3} is IN. If L1, L2 or L3 became OUT (instead of L4), the polygon would still become OUT and a polyline formed by the remaining segments would become IN. In step 4, L4 is restored, and the polygon that contained L4 becomes IN again. The previous polyline containing L1, L2 and L3 becomes OUT because it is a subset of the polygon.

4.3 Supporting High-Level Inferences

Along with maintaining the set of spatial relations, the LLRD also supports the HLRD's high-level reasoning. The LLRD continually provides a correct set of spatial relations for the HLRD. In addition, when relations change in the LLRD's dependency network, these changes are propagated directly to the HLRD. The HLRD then changes its diagram interpretation accordingly.

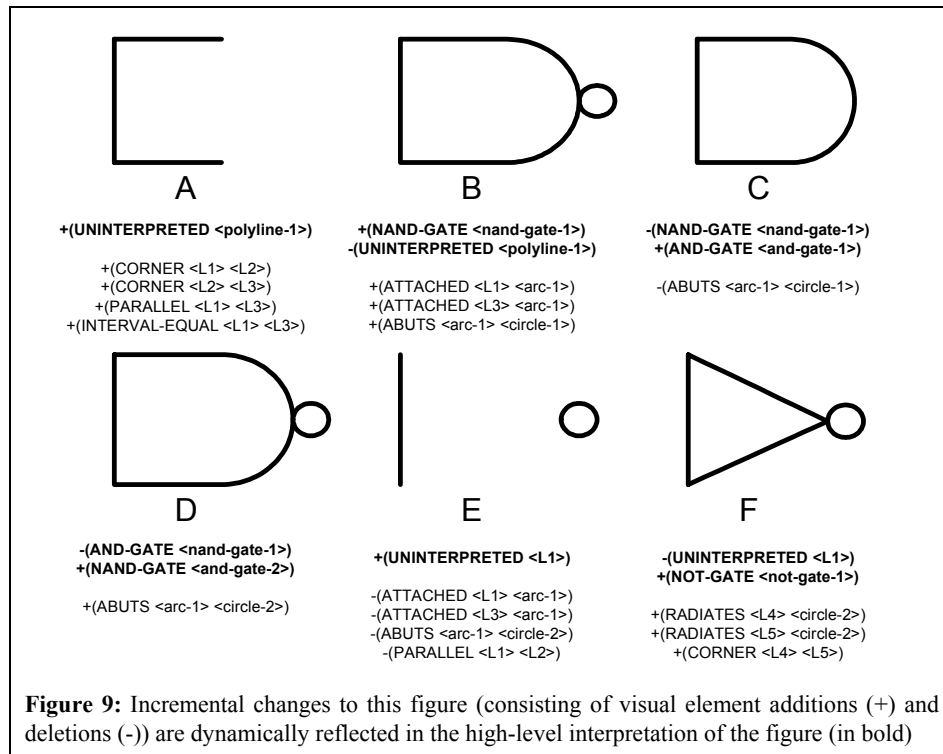
Figure 9 shows an example of how small visual changes can dynamically change the diagram interpretation. Here, gradual additions and deletions to a diagram change the diagram from an uninterpreted figure (A), to a NAND gate (B), then to an AND gate (C), back to a NAND gate (D), and then to a NOT gate (F). At each point, the LLRD's dependency network manages the set of spatial relations that are available to the HLRD. The HLRD, in turn, modifies its representation automatically as these changes are made to the figure.

To make this work, the internal relations of nodes in the LLRD's dependency network are linked with logic nodes (each representing a specific visual relation) in the HLRD's LTMS. If a node in the LLRD is retracted or if the selected internal relation is changed, these changes are propagated to the LTMS. Changes to an LTMS node's truth value automatically triggers the Boolean Constraint Propagation algorithm [20] to update the LTMS's belief state.

The LTMS is a more powerful reasoner than the LLRD's dependency network, but the network is still an adequate foundation for the LTMS given the constraints of spatial domains. An LTMS make inferences based on both true and false nodes, while the LLRD's dependency network is roughly equivalent to the nodes in a Justification-based TMS (JTMS). Such nodes represent only Horn clauses, and do not distinguish between facts that are false and those that are unknown. However, for any JEPD set represented by a LLRD node, we can make a closed-world assumption over the set of internal relations that allows us to treat the selected internal relation as true, and the rest as false. In addition, due to the nature of visual relation detection, when an LLRD node is OUT, all of its internal relations can be treated as false, and not simply unknown. This is because an LLRD node becomes OUT only when a necessary visual precondition becomes invalid.

4.4 Handling Default Assumptions

Finally, to allow the HLRD to properly handle incremental LLRD information, we extended the default assumption mechanism in the HLRD's LTMS (Figure 10). While the



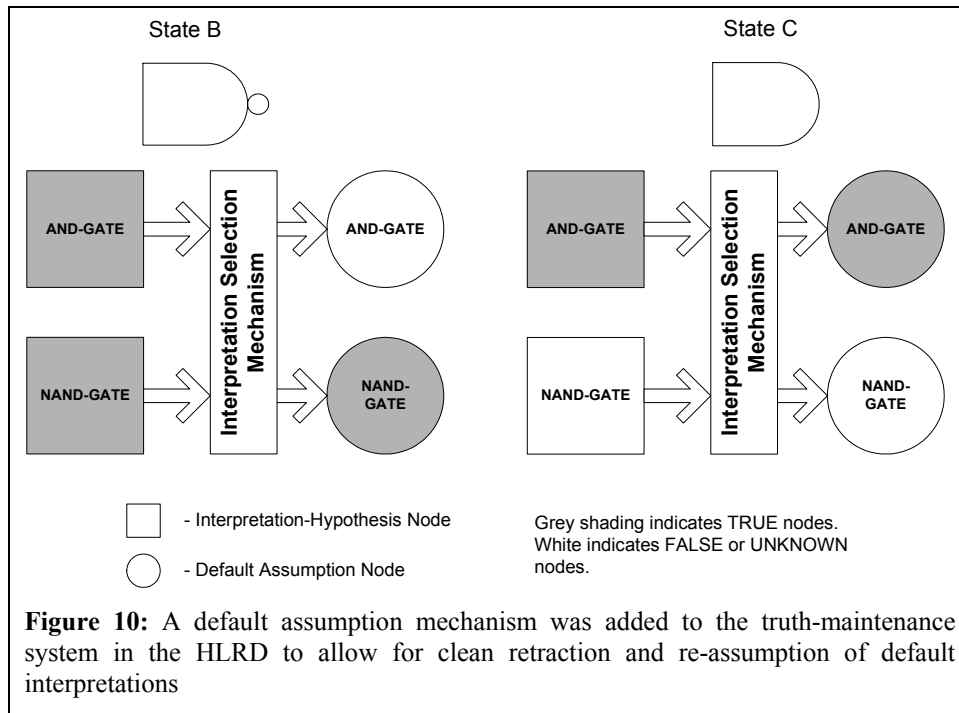
LTMS already supports simple incremental reasoning, it does not support dynamically changing the high-level interpretation when it depends on default reasoning.

For example, in Figure 9(B), the current visual domain theory supports an interpretation of the figure as both a NAND gate as well as an AND gate. Both interpretations are assumed, and when they are found to be in conflict, the AND gate interpretation is retracted.

This system works well for a diagrammatic reasoner that functions in batch mode, where retracted assumptions do not need to be re-examined. In an incremental reasoner, however, the visual elements and relations that lead to an over-ruled default interpretation may themselves change. When the circle is removed as in Figure 9(C), the standard LTMS assumption mechanism cannot re-assume the AND interpretation because it has already been explicitly retracted.

To handle this problem, the new default assumption mechanism uses two additional node types: a *default assumption node* and an *interpretation-hypothesis node*. The default assumption node is an extension to the existing LTMS node structure, with slots added to store alternative interpretations and to link to an interpretation-hypothesis node. The interpretation-hypothesis node is a standard LTMS node, created as an assumption and justified by the appropriate implicational structure.

These nodes allow all potential interpretations to be available even if some have been previously rejected. When a new composite object is detected, it is *default-assumed*, rather than assumed. This creates both a default assumption node for the new



object as well as an interpretation-hypothesis node. Thus, for each potential interpretation, there will be a valid interpretation-hypothesis node. However, at any instance, there only one valid interpretation exists for each set of interpretation hypotheses. The existence of multiple interpretation-hypothesis nodes causes a contradiction within the LTMS, triggering the interpretation selector.

Selection between potential interpretations is handled in a domain-dependent fashion. For example, in the logic circuit domain, the maximally-preferred alternative is the interpretation with the most elements [21]. In the example, the NAND gate would be selected because the AND gate is a subset.

This handles the case where one composite object has two possible interpretations. However, to support dynamically changing interpretations, we must also consider what happens when removing part of an object requires revising our interpretation again. In this case, we may want to revert to a previous interpretation that was discarded because it was not the maximally preferred.

This is handled by activating the selector mechanism when an interpretation hypothesis node is retracted. In the example, this corresponds to removing the circle from the NAND gate, which causes a retraction of the NAND gate interpretation, and the reactivation of the AND gate interpretation. The interpretation selector returns to the next-best alternative interpretation, allowing the AND gate interpretation (and all its high-level consequences) to be reactivated.

5 Integrating Incremental GeoRep into a Drawing Program

We use an existing drawing system, JFig [22], as an interface to GeoRep. JFig is a Java implementation of the well-known XFig program [23], and is freely available on the web. We customized the JFig interface (Figure 11) to notify GeoRep of visual element adds, deletes, and modifies.

Whenever a new element is added or deleted from the diagram in JFig, the corresponding element in GeoRep is added or removed, and the dependency network is updated. The restore command works on the most recently removed object, and makes the corresponding element valid again in GeoRep.

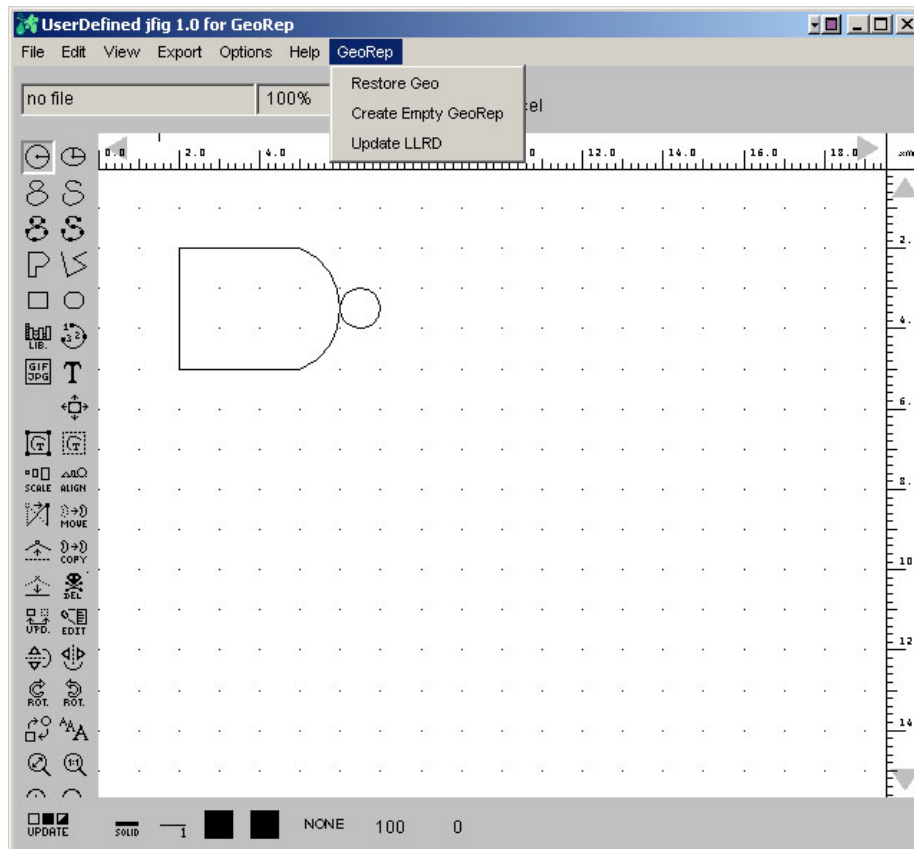


Figure 11: Drawing a figure in JFIG. An additional menu allows the interface to control the link to the incremental version of GeoRep

6 Discussion

We have presented an extensible framework for incremental reasoning over a variety of spatial relations. This framework employs jointly exhaustive and pairwise disjoint relations to encapsulate visual reasoning subtasks.

Although the dependency network handles the addition, retraction, and restoration of visual elements, there are many ways in which the dependency network could be used to make more powerful inferences. For example, it could aid in the re-evaluation of modified visual elements. Because each relation node depends on a procedural visual test to choose between its alternative internal relations, once a visual element is modified, the tests must be rerun to determine if the previously chosen internal relation remains valid. If not, a chain of visual tests must be applied to the modified visual element and all proximate elements.

Another potential use of this network is to let the HLRD influence the LLRD by, for example, setting test tolerance values (e.g., the margin for a perpendicular line test). For example, it should be possible to have the HLRD detect quadrilaterals that are almost square, and then modify the *relative-angle* test so that the LLRD recognizes a necessary corner as perpendicular.

Finally, there is the problem of tradeoffs between spatial reasoning and visual processing. Assuming that visual processing is extremely cheap, which spatial relations are really worth caching in this kind of mechanism? In this architecture, we have assumed that even very low-level spatial relations are worth caching in order to test the implications of the architecture. Clearly, however, the utility of caching these relations depends critically on the task and the power of the visual processing system (i.e., the visual operators available to the LLRD).

One other spatial reasoning task that is especially important in geographic reasoning is proper treatment of scaling. Geographic elements or features that are critical at one scale (e.g., sidewalks) may be of little or no interest at larger scales.

Although it would be interesting to discuss how GeoRep might handle scale issues (for example, sets of visual elements could be turned into composite glyphs that would limit processing on its internal elements), GeoRep currently deals with scaling issues through the HLRD's visual domain theory, which means, inevitably, that scale is handled in a domain-dependent fashion.

This provides GeoRep with a lot of flexibility in handling scaling issues. For example, in the COA domain (Figure 5), military units (which are rectangular) have visual extent, but are treated as point objects by the visual domain theory. Other geographic elements, such as "Objective Buford," scale with the picture, since the extent is meaningful geographically.

Finally, we note that this work is only one part of a larger effort to create a next-generation diagrammatic reasoner. This reasoner will combine incremental spatial reasoning with other abilities, such as dynamic reinterpretation of diagrams.

Acknowledgements

We would like to thank Norman Hendrich, the creator of JFig, for very useful advice and suggestions on interfacing to that program. We would also like to thank Ken Forbus and several anonymous reviewers for helpful suggestions.

References

1. Ferguson, R.W., and K.D. Forbus. 2000. GeoRep: A flexible tool for spatial representation of line drawings. *In* Proceedings of the 18th National Conference on Artificial Intelligence. AAAI Press, Austin, Texas. 510-516.
2. Hernández, D. 1993. Maintaining qualitative spatial knowledge. *In* Proceedings of the European Conference on Spatial Information Theory (COSIT), Elba, Italy. 19-22.
3. Freksa, C. 1992. Temporal reasoning based on semi-intervals. *Artificial Intelligence* 54:199-227.
4. Egenhofer, M., and K. Al-Taha. 1992. Reasoning about gradual changes of topological relationships. *In* Theory and Methods of Spatio-Temporal Reasoning in Geographic Space. A. Frank, I. Campari, and U. Formentini, editors. Springer-Verlag, Pisa, Italy. 196-219.
5. Davis, R. 2002. Position statement and overview: Sketch recognition at MIT. *In* AAAI Spring Symposium on Sketch Understanding, Palo Alto, CA.
6. Scher, D. 2000. Lifting the curtain: The evolution of the Geometer's Sketchpad. *The Mathematics Educator* 10:42-48.
7. Ferguson, R.W., R.A.J. Rasch, W. Turmel, and K.D. Forbus. 2000. Qualitative spatial interpretation of Course-of-Action diagrams. *In* Intelligent Systems Demonstrations, 18th National Conference on Artificial Intelligence, Austin, Texas.
8. Ferguson, R.W., and K.D. Forbus. 1998. Telling juxtapositions: Using repetition and alignable difference in diagram understanding. *In* Advances in Analogy Research. K. Holyoak, D. Gentner, and B. Kokinov, editors. New Bulgarian University, Sofia, Bulgaria. 109-117.
9. Ferguson, R.W. 2001. Symmetry: An Analysis of Cognitive and Diagrammatic Characteristics. Ph.D. Dissertation, Department of Computer Science, Northwestern University. Evanston, Illinois.
10. Ferguson, R.W. 2000. Modeling orientation effects in symmetry detection: The role of visual structure. *In* Proceedings of the 22nd Conference of the Cognitive Science Society. Erlbaum, Hillsdale, New Jersey. 143.
11. Ferguson, R.W., A. Aminoff, and D. Gentner. 1996. Modeling qualitative differences in symmetry judgments. *In* Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society. Lawrence Erlbaum Associates, Hillsdale, NJ. 12.
12. Hoffman, D.D., and W.A. Richards. 1984. Parts of recognition. *Cognition* 18:65-96.
13. Rock, I. 1973. Orientation and Form. Academic Press, New York, NY.

14. Allen, J.F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26:832-843.
15. Ferguson, R.W. 1994. MAGI: Analogy-based encoding using symmetry and regularity. *In Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*. A. Ram and K. Eiselt, editors. Lawrence Erlbaum Associates, Atlanta, GA. 283-288.
16. Forbus, K.D., and J. de Kleer. 1993. *Building Problem Solvers*. The MIT Press, Cambridge, MA.
17. Cohn, A.G. 1997. Qualitative spatial representation and reasoning techniques. *In Proceedings of KI-97*. G. Brewka, C. Habel, and B. Nebel, editors. Springer-Verlag, Freiburg, Germany. 1-30.
18. Cohn, A.G., D.A. Randell, Z. Cui, and B. Benett. 1993. Qualitative spatial reasoning and representation. *In Proc of the IMACS Workshop on Qualitative Reasoning and Decision Technologies, QUARDET '93*. N.P. Carrete and M. Singh., editors. CIMNE, Barcelona. 513-522.
19. Cohn, A.G. 1995. A hierarchical representation of qualitative shape based on connection and convexity. *In International Conference on Spatial Information Theory COSIT-95*, Semmering, Austria.
20. McAllester, D. 1990. Truth maintenance. *In Proceedings of the Eighth National Conference on Artificial Intelligence*. AAAI Press, Boston, MA. 1109-1116.
21. Doyle, J. 1992. Rationality and its roles in reasoning. *Computational Intelligence* 8:376-409.
22. Hendrich, N. 1999. JavaFIG: The Java diagram editor. Computer Science Department, University of Hamburg, Germany.
23. Smith, B.V. 1999. XFig. <http://www.xfig.org>