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
801 Atlantic Avenue
Atlanta, GA 30332
{rwf, jlbokor, cmappus, storm} @cc.gatech.edu

Abstract

Because diagrams are often created incrementally, a qualitative diagrammatic reasoning system must dynamically manage a potentially large set of spatial interpretations. This paper describes an architecture for 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. Examples of these spatial relation sets are interval relations, relative orientation relations, and connectivity relations. The network 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. We also describe how the system supports high-level reasoning, including support for creating default assumptions. Finally, we show 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 across a wide variety of reasoning tasks. Because a single diagram conveys many spatial relations at a glance, it provides a rich medium for many domains, including geography, architecture, and engineering. A body of research exists showing how diagrams are used in many cognitive tasks (Glasgow, Narayanan, & Chandrasekaran, 1995).

These characteristics are more interesting when we consider that the spatial relations in a diagram may not be static. Diagrams frequently change over time. The addition, removal, and modification of elements also changes 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.

An incremental processing approach is also be motivated other factors. For example, problem solving may drive changes to a particular diagram as new ideas or subtasks emerge. Also, if the conceptual interpretation of the spatial relations is computationally expensive, we do not want to recompute the interpretation when only minor changes are made. Finally, a more practical benefit of incremental processing is that it distributes the processing burden 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 the maintenance of incremental, nonmonotonic changes to a diagram. This work extends the GeoRep diagrammatic reasoner (Ferguson & Forbus, 2000), which is described in section 3. After describing GeoRep, we discuss how GeoRep was modified to allow incremental processing, and cover a number of implementation issues: how to handle composite objects, the interface between low-level and high-level reasoning, and a modified default assumption mechanism. We also describe extensions to a user interface allowing a user to create diagrams and update the inferences of the reasoner. We then discuss future challenges for this architecture.

2. Related Work

In qualitative spatial reasoning, researchers have explored how to process qualitative spatial vocabularies incrementally. Notably, Hernández (1993) proposed mechanisms for maintaining consistent spatial knowledge that include a propagation heuristic for inserting relations and reason maintenance mechanisms for deleting relations. These mechanisms use a dependency network, similar to that in a truth-maintenance system. The system uses a combined set of orientation and topological spatial relations to represent qualitative 2-D positions.

Other research has explored the use of conceptual neighborhoods (Freksa, 1992) and topological distances (Egenhofer & Al-Taha, 1992) 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 (Davis, 2002) 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.

Another approach is found in the Geometer's Sketchpad, which uses a constraint system to dynamically enforce a set

of constraints over abstract geometric elements, including lines, rays, and circles (Scher, 2000). These constraints include geometric relations such as the attachment of a segment endpoint along a circle perimeter and the bisection of an angle. The user can then move points in the figure as desired, subject only to the geometric constraints. This allows visualization of simple geometric conjectures, such as the (true) conjecture that the bisectors of a triangle's angles intersect at a single central point. However, unlike those in GeoRep, the spatial relations in Geometer's Sketchpad are not discovered by the system, but are entered explicitly by the user.

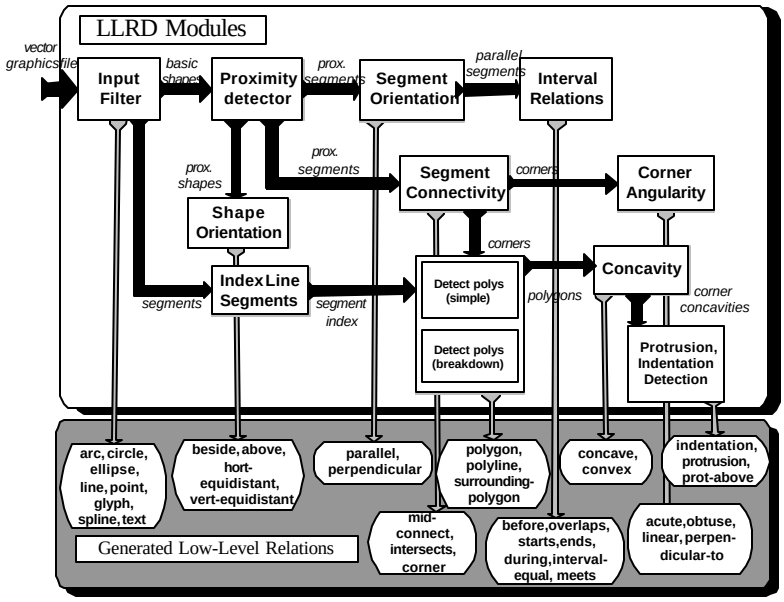


Figure 2: Data flow diagram of GeoRep's Low-Level Relation Describer (LLRD). The top portion shows the collections of visual operators and the visual elements they process. The lower portion shows the set of spatial relations produced by each set of visual operators.

3. The GeoRep Reasoner

This work extends an existing diagrammatic reasoner, GeoRep, to make processing incremental. GeoRep (Ferguson & Forbus, 2000) has been used in a number of diagrammatic reasoning domains, including military diagrams (Ferguson, Rasch, Turmel, & Forbus, 2000), simple physical diagrams (Ferguson & Forbus, 1998), and logic circuits (Ferguson, 2001). In addition, it has been used as the visual representation system for several cognitive modeling simulations (Ferguson, 2000; Ferguson, Aminoff, & Gentner, 1996).

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

To generate this representation, GeoRep uses a two-stage architecture (Figure 1).

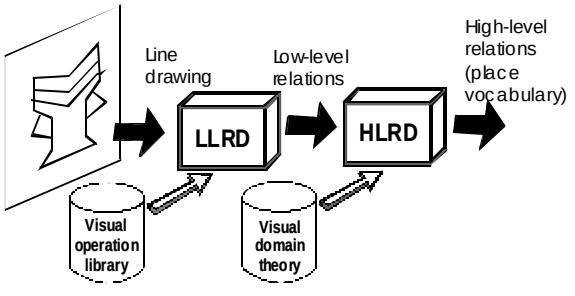


Figure 1: Simplified GeoRep architecture, containing stages for low-level and high-level visual reasoning

The first stage, the *Low-Level Relational Describer* (LLRD; Figure 2) represents a set of low-level visual relations. These low-level relations are generated starting with proximity calculations and ending with more complex 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 figure boundaries (Hoffman & Richards, 1984), and to vertical and horizontal orientations in the assumed frame of reference (Rock, 1973). Interestingly, one relation set used that has not been tested for early vision is interval relations (Allen, 1983) between parallel lines. In practice, these relations are extremely useful in modeling aspects of visual perception such as the detection of qualitative symmetry (Ferguson, 1994). 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, and multiple LLRDs can be used to simulate visual separation based on preattentive factors such as color (Ferguson, 2001).

The second stage, the *High-Level Relational Describer* (HLRD; Figure 3) uses these low-level relations and a rule-

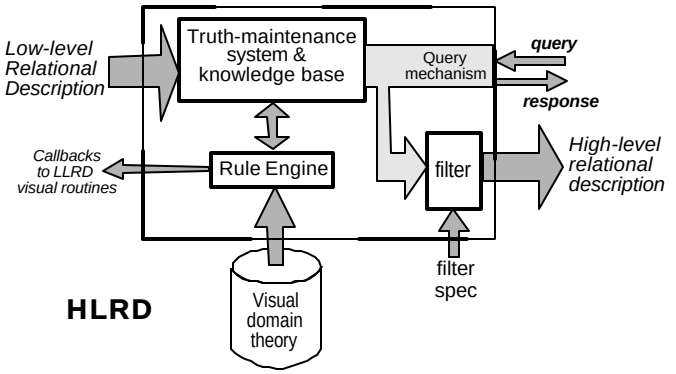


Figure 3: The HLRD architecture, which reasons over the low-level description to produce a domain-specific interpretation

based *visual domain theory* to produce a description of the diagram. The output of the HLRD describes the diagram using domain-dependent high-level relations. For example, using domain-dependent rules, the HLRD produces the description of the logic circuit in Figure 4, which includes 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.

The HLRD rules use a pattern-directed inference system that is supported by a logic-based truth maintenance system (LTMS) (Forbus & de Kleer, 1993; McAllester, 1990). HLRD rules use the LLRD's low-level visual relations as well as domain knowledge, such as an ontology of logic gate types. HLRD rules are typically constrained to run only on proximate visual elements and can call visual operations within the LLRD.

Once the HLRD has generated a high-level description, it can either be retrieved directly, or filtered by relation type to simulate different diagrammatic representation levels (Ferguson & Forbus, 2000).

4. Making GeoRep Incremental

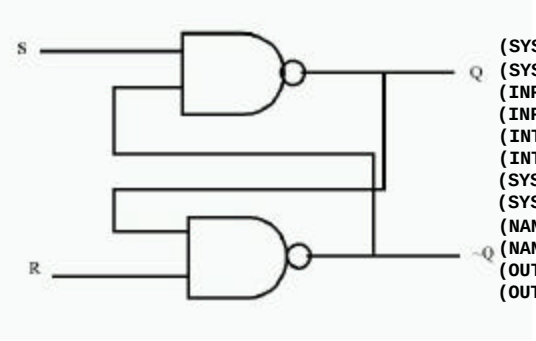
One limitation of GeoRep is its processing model. GeoRep processes figures in batch mode on static diagrams. This is due to limitations of the LLRD rather than the HLRD. The HLRD is inherently incremental because it is based on an LTMS. Visual relations can be assumed or retracted at any time, and the 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. The LLRD does not store information about which visual elements are used in particular spatial relations.

Therefore, making GeoRep incremental requires an incremental LLRD. The following sections describe how we modified the LLRD and extended GeoRep to process diagrams incrementally.

4.1 Creating the LLRD dependency network

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) (Cohn, 1997). 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 (Allen, 1983) and RCC (Cohn, 1995; Cohn, Randell, Cui, & Benett, 1993). 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.

The incremental LLRD includes a dependency network (Figure 5) that tracks the low-level relations supported by each visual element, maintaining consistency as visual elements are added or removed and storing the necessary information to allow reconsideration when elements are modified.




```
(SYSTEM-OUTPUT WIRE-S11)
(SYSTEM-OUTPUT WIRE-S15)
(INPUT-LABEL <POSITIONED-TEXT-3> WIRE-S1 "S")
(INPUT-LABEL <POSITIONED-TEXT-4> WIRE-S10 "R")
(INTERNAL-CONNECTION WIRE-12)
(INTERNAL-CONNECTION WIRE-11)
(SYSTEM-INPUT WIRE-S10)
(SYSTEM-INPUT WIRE-S1)
(NAND-GATE NAND-GATE-2)
(NAND-GATE NAND-GATE-1)
(OUTPUT-LABEL <POSITIONED-TEXT-1> WIRE-S11 "~Q")
(OUTPUT-LABEL <POSITIONED-TEXT-2> WIRE-S15 "Q")
```

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

Each relation node in the network has five key parts: a relation type, a set of internal relations, a selected internal relation, antecedents and consequences. Each node also has a set of internal relations based on its relation type.

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 5 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 (Allen, 1983). The propagation algorithm is similar to truth-value propagation in a justification-based truth maintenance system (Forbus & de Kleer, 1993).

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.

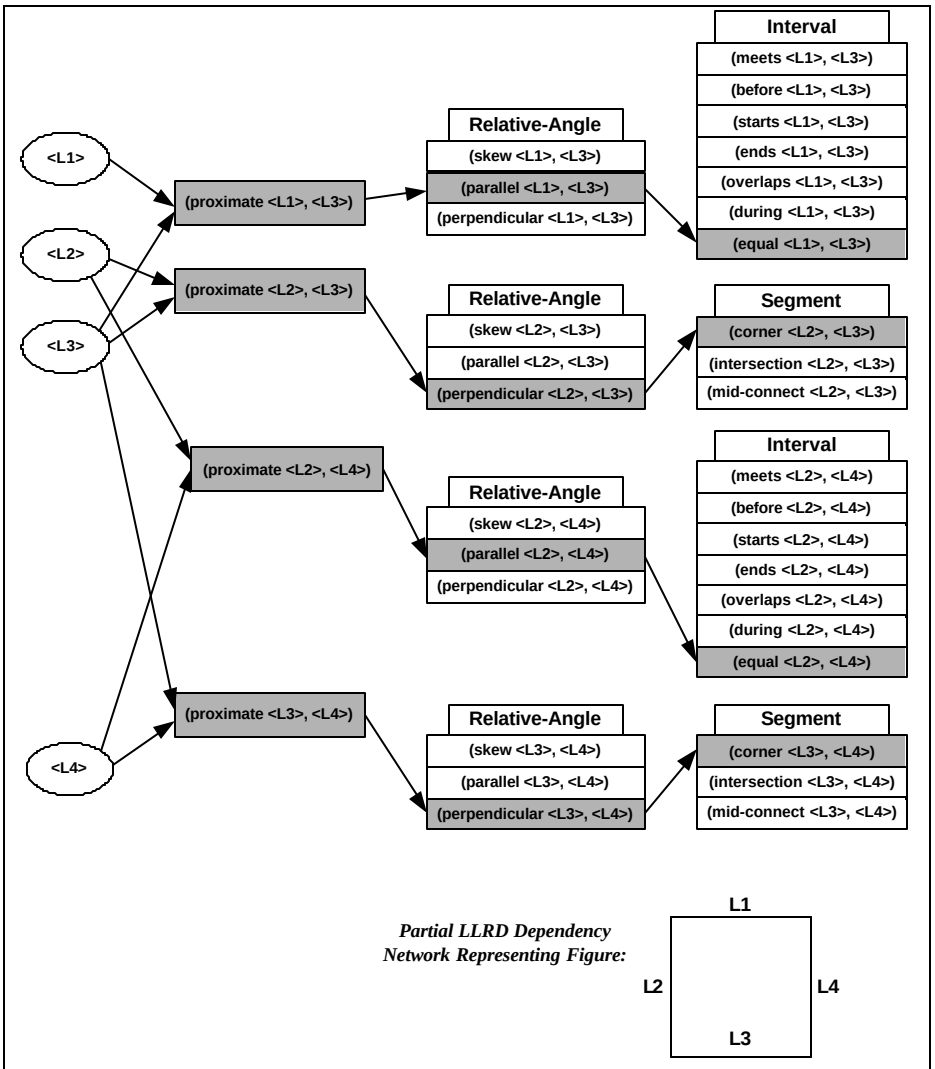


Figure 5: Subsection of the dependency network. The selected internal relation for each node is shaded.

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.

In contrast to a JTMS, it is important to note that the truth value of the network is not a direct function of the truth values of the antecedent 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 L3, we can decompose the set of relations using the dependency network as follows:

interval-equal (L1,L3) ≡
  parallel(L1,L3) ∧ *interval-test*(L1,L3,interval-equal).
parallel(L1,L3) ≡
  proximate(L1,L3) ∧ *relative-angle-test*(L1,L3,parallel).
proximate(L1,L3) ≡ L1 ∧ L3 ∧ *proximate-test*(L1,L3).

Therefore:

interval-equal(L1,L3) ↔
  L1 ∧ L3 ∧ *proximate-test*(L1,L3)
    ∧ *relative-angle-test*(L1,L3,parallel)
    ∧ *interval-test*(L1,L3,interval-equal).

This is equivalent to the set of tests applied to segments in the original 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.

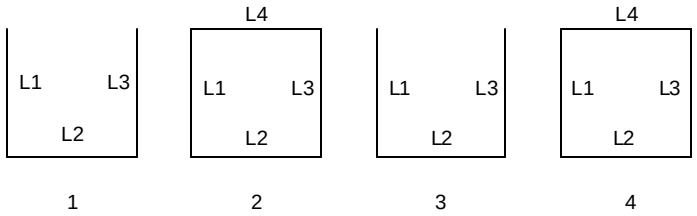
4.2 Handling composite objects

Composite objects (objects composed of multiple visual elements, such as polylines and polygons) are a special problem in incremental spatial reasoning.[1] 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.

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 removed 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.



Polyline/Polygon Detection Example:
Step 1 A polyline is detected (polyline <L1> <L2> <L3>).
Step 2 A new line <L4> is added, and it as well as its proximate neighbors <L1>, <L2>, <L3> are added to the list of elements to process. The polyline element is marked OUT and a new polygon element (polygon <L1> <L2> <L3> <L4>) is added.
Step 3 The most recently added line <L4> is removed, the polygon is marked OUT and the remaining lines are processed. The previous polyline is detected and marked IN.
Step 4 The removed line <L4> is restored and marked IN, and its component elements are processed, marking the polyline OUT and the polygon IN.

Figure 6: Handling polylines and polygons in the incremental LLRD

An example of how the LLRD handles composite objects is given in Figure 6. 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 (becomes OUT), and the polygon becomes 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 so that it can change its diagram interpretation accordingly.

Figure 7 shows an example of how small visual changes can dynamically change the diagram interpretation. Here, gradual additions and deletions to a diagram change the

[1] 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 (Ferguson & Forbus, 1998). 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.

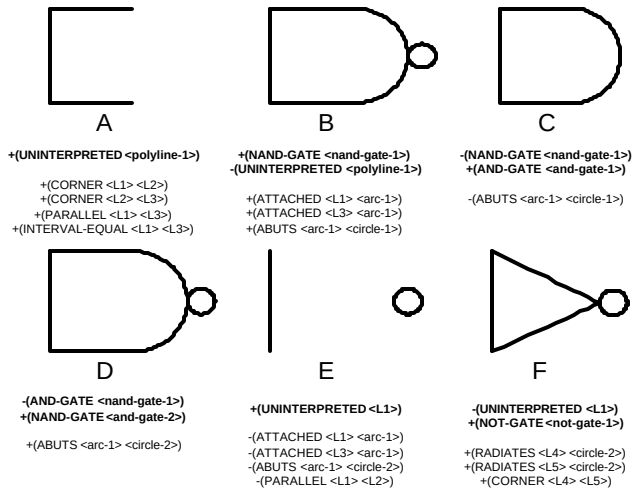


Figure 7: Incremental changes to this figure (consisting of visual element additions (+) and deletions (-)) are dynamically reflected in the high-level interpretation of the figure (in bold).

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.

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 (McAllester, 1990) 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 makes 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 unlike LTMS nodes, do not distinguish between false and 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 8). 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 7 (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 7(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.

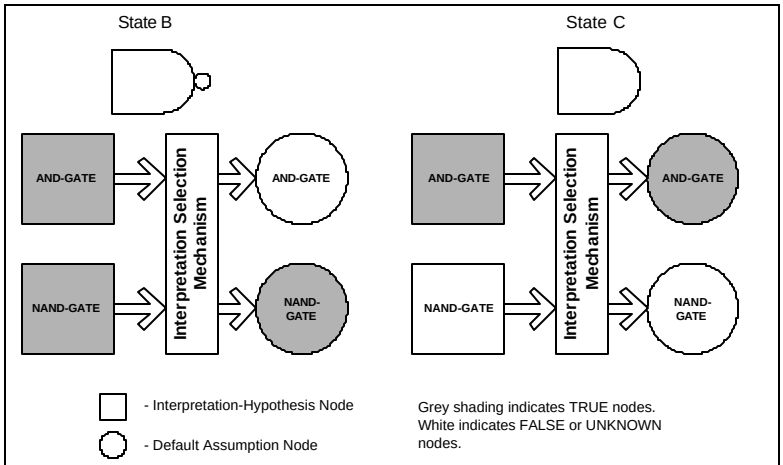


Figure 8: A default assumption mechanism was added to the truth-maintenance system in the HLRD to allow for clean retraction and re-assumption of default interpretations.

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, 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 (Doyle, 1992) is the interpretation with the most elements. In the example, the NAND gate is 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. Integration into a Drawing Program

We use an existing drawing system, JFig (Hendrich, 1999), as an interface to the reasoner. JFig is a Java implementation of the well-known XFig program (Smith, 1999), and is freely available on the web. We customized the JFig interface (Figure 9) to notify the reasoner 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.

6. Conclusions and Future Work

We have presented an extensible architecture 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).

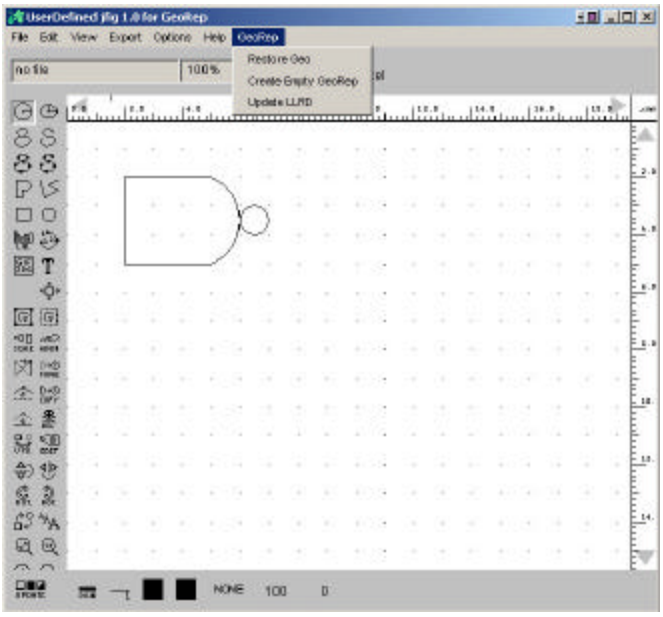


Figure 9: Drawing a figure in JFIG. An additional menu allows the interface to control the link to the incremental version of GeoRep.

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

An earlier version of this paper will appear at the 2003 Conference on Spatial Information Theory. Portions of this research were supported by the Cognitive Science and Computer Science programs of the Office of Naval Research, and by the National Science Foundation under the Learning and Intelligent Systems program. We would also like to thank Norman Hendrich, the creator of JFig, for very useful advice and suggestions on interfacing to that program. Special thanks to Ken Forbus and several anonymous reviewers for feedback on an earlier version of this paper.

References

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM, 26*, 832-843.

Cohn, A. G. (1995). A hierarchical representation of qualitative shape based on connection and convexity, *International Conference on Spatial Information Theory COSIT-95*. Semmering, Austria.

Cohn, A. G. (1997). Qualitative spatial representation and reasoning techniques. In G. Brewka & C. Habel & B. Nebel (Eds.), *Proceedings of KI-97* (pp. 1-30). Freiburg, Germany: Springer-Verlag.

Cohn, A. G., Randell, D. A., Cui, Z., & Benett, B. (1993). Qualitative spatial reasoning and representation. In N. P. Carrete & M. Singh. (Eds.), *Proc of the IMACS Workshop on Qualitative Reasoning and Decision Technologies, QUARDET '93* (pp. 513-522). Barcelona: CIMNE.

Davis, R. (2002). *Position statement and overview: Sketch recognition at MIT.* Paper presented at the AAAI Spring Symposium on Sketch Understanding, Palo Alto, CA.

Doyle, J. (1992). Rationality and its roles in reasoning. *Computational Intelligence, 8*(2), 376-409.

Egenhofer, M., & Al-Taha, K. (1992). Reasoning about gradual changes of topological relationships. In A. Frank & I. Campari & U. Formentini (Eds.), *Theory and Methods of Spatio-Temporal Reasoning in Geographic Space* (Vol. 639, pp. 196-219). Pisa, Italy: Springer-Verlag.

Ferguson, R. W. (1994). MAGI: Analogy-based encoding using symmetry and regularity. In A. Ram & K. Eiselt (Eds.), *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society* (pp. 283-288). Atlanta, GA: Lawrence Erlbaum Associates.

Ferguson, R. W. (2000). Modeling orientation effects in symmetry detection: The role of visual structure, *Proceedings of the 22nd Conference of the Cognitive Science Society* (pp. 143). Hillsdale, New Jersey: Erlbaum.

Ferguson, R. W. (2001). *Symmetry: An Analysis of Cognitive and Diagrammatic Characteristics.* Unpublished Ph.D., Northwestern University, Evanston, Illinois.

Ferguson, R. W., Aminoff, A., & Gentner, D. (1996). Modeling qualitative differences in symmetry judgments, *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society* (pp. 12). Hillsdale, NJ: Lawrence Erlbaum Associates.

Ferguson, R. W., & Forbus, K. D. (1998). Telling juxtapositions: Using repetition and alignable difference in diagram understanding. In K. Holyoak & D. Gentner & B. Kokinov (Eds.), *Advances in Analogy Research* (pp. 109-117). Sofia, Bulgaria: New Bulgarian University.

Ferguson, R. W., & Forbus, K. D. (2000). GeoRep: A flexible tool for spatial representation of line drawings, *Proceedings of the 18th National Conference on Artificial Intelligence* (pp. 510-516). Austin, Texas: AAAI Press.

Ferguson, R. W., Rasch, R. A. J., Turmel, W., & Forbus, K. D. (2000). Qualitative spatial interpretation of Course-of-Action diagrams, *Proceedings of the 14th International Workshop on Qualitative Reasoning*. Morelia, Mexico.

Forbus, K. D., & de Kleer, J. (1993). *Building Problem Solvers*. Cambridge, MA: The MIT Press.

Freksa, C. (1992). Temporal reasoning based on semi-intervals. *Artificial Intelligence, 54*(1-2), 199-227.

Glasgow, J., Narayanan, N. H., & Chandrasekaran, B. (1995). *Diagrammatic Reasoning: Cognitive and Computational Perspectives*. Menlo Park, CA: The AAAI Press/The MIT Press.

Hendrich, N. (1999). *JavaFIG: The Java diagram editor* [Web page]. Computer Science Department, University of Hamburg, Germany. Retrieved December 28, 1999, from the World Wide Web: http://tech1.informatik.uni-hamburg.de/applets/javafig/

Hernández, D. (1993). Maintaining qualitative spatial knowledge, *Proceedings of the European Conference on Spatial Information Theory (COSIT)* (pp. 19-22). Elba, Italy.

Hoffman, D. D., & Richards, W. A. (1984). Parts of recognition. *Cognition, 18*(1-3), 65-96.

McAllester, D. (1990). Truth maintenance, *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 1109-1116). Boston, MA: AAAI Press.

Rock, I. (1973). *Orientation and Form*. New York, NY: Academic Press.

Scher, D. (2000). Lifting the curtain: The evolution of the Geometer's Sketchpad. *The Mathematics Educator, 10*(2), 42-48.

Smith, B. V. (1999). XFig: http://www.xfig.org.