A Diagrammatic Reasoning Architecture: Design, Implementation and Experiments

B. Chandrasekaran, Unmesh Kurup and Bonny Banerjee

Department of Computer Science and Engineering
The Ohio State University
Columbus, OH 43210
{Chandra|Kurup|Banerjee}@cse.ohio-state.edu

Abstract

This paper explores the idea that the cognitive state during problem solving diagrams is bi-modal, one of whose components is the traditional predicate-symbolic representation composed of relations between entities in the domain of interest, while a second component is an internal diagrammatic representation. In parallel with the operators in the symbolic representation that are based on symbol matching and inferencing, there is a set of operators in the diagrammatic component that apply perceptions to the elements of the diagram to generate information. In addition there is a set of diagram construction operations that may modify the diagram by adding, deleting and modifying the diagrammatic elements, in the service of problem solving goals. We describe the design of the diagrammatic component of the architecture, and show how the symbolic and diagrammatic modes collaborate in the solution of a problem. We end the paper with a view of the cognitive state as multi-modal, in consonance with our own phenomenal sense of experiencing the world in multiple modalities and using these senses in solving problems.

Bimodality of the Problem State in Diagrammatic Reasoning

The standard account of cognitive state in both AI and cognitive science involves state representations that are sentences composed of predicates that describe relations that hold between various entities in the domain of interest[1]. Cognitive state changes as a result of applying operators that change parts of the descriptions into other descriptions. For example, an agent contemplating a Blocks World situation would have state representations of the form ON(A, B) ^ ON(C, D), and the state change corresponding to removing block A to the table would be accomplished by Add & Delete operators that change the description to ON(A, Table) ^ ON(C, D). If for his problem solving goals he needs to know the Left-of or Right-of relation between the blocks at various times, as each operation is carried out a number of Left-of and Right-of relations would need to be added and deleted. For even a modest number of blocks, the number of such relations to be updated can become significant in number.

Now suppose the agent is now solving the same Blocks world problem while looking at actual blocks. As he moves Block A to the table, there is really no need for an internal set of Add and Delete rules relating to relations between the blocks. As the movement is done, he can simply look at the blocks and note the relations that are relevant for that stage in problem solving. The problem state can now be conceived as *bimodal*, a part that corresponds to traditional predicate-symbolic representations, and another part that is spatial and available for perception. Extraction of information such as Left-of(B,C) can be thought of as applying a perception operator to the spatial component of the problem state as a parallel to the application of symbolic rule-based operators to the symbolic component of the problem state.

Use of diagrams in problem solving is similar. When we have a diagram of a situation, all spatial consequences of an operation, such that of adding or deleting a diagrammatic element or changing the spatial properties of the elements, do not need to be inferred by application of rules to the symbolic component, but obtained as needed from the diagram by applying perception or direct measurement operations on the diagram. The diagram automatically encodes certain consequences of changes, such as *emergent objects* and *emergent relations*. When two roads are diagrammed by two curves in a diagram and they intersect, the intersection point and the segments are emergent objects of representational interest. Similarly, when a block A is placed to the left of B which is already to the left of C, an emergent relation is that A is to the left

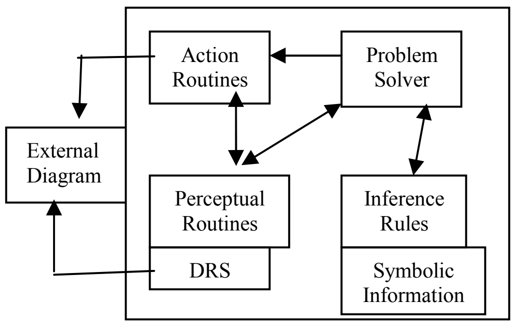
[1] This symbolic representation framework applies to both logicist traditions in AI as well as those schools, such as frame and script theories, that situate themselves as alternatives to logicism. The main idea is that all of them view the agent as having knowledge that is expressed in terms of symbols that stand for individuals and relations between them. The differences between the logicists and logic skeptics are about content theories and inference, not about the predicate structure of the representation.

of C. Perception picking up the spatial consequences of changes has been called a *free ride* (Shimojima, 1996) and is one of most important advantages of a diagrammatic representation.

Diagrams do not have to be external. In a good deal of human problem solving people experience mental images similar to a diagram. Of course internal diagrams are subject to short term memory limitations, but phenomenologically, problem solving has a character similar to that when external diagrams are involved. That is, they apply what they experience as internal perception and extract some information that they may combine with linguistically couched information to make further inferences. In these cases, the problem state is completely internal and bimodal.

We have been developing an architecture for diagrammatic reasoning that is intended to support problem solving with such bimodal problem states. The linguistic/symbolic mode supports knowledge representations and inferences that are traditional in AI, while the diagrammatic mode supports a representation that captures the spatial aspects of the configuration of objects in a diagram, and an open-ended set of perceptions that can add information to the symbolic component of the problem state. By the same token, there are operators that can modify the diagram in various ways to capture the effects of intended operations. Problem solving is goal-driven, as in cognitive architectures such as Soar. Problem solving proceeds *opportunistically*, whichever mode can help in the solution of a subtask makes the contribution, and the problem state changes. This process repeats until the problem is solved (assuming it is solvable given the knowledge). The goal of this paper is to outline our architecture, detail some aspects of the diagrammatic representation, and give an example of its use.

The Architecture



DRS (for diagrammatic representation system) is the internal representation of the diagram, thus providing the diagrammatic part of the bimodal problem state. DRS can be the result of perception on an external image of a diagram, (constructed by that part of visual perception that breaks the image array into objects, or manually, as in our experiments), composed internally from perceptual memory fragments, or a combination of the two. Thus the architecture is a functional diagrammatic problem solver even in the absence of an external diagram. DRS is intended to correspond to the experience of seeing a diagram as a configuration of spatial objects. A set of perceptions can generate information about the diagram. Action routines construct or modify aspects of the diagram to satisfy various constraints. The problem solver treats both the diagrammatic and the symbolic components as parallel representations and potentially equal sources of progress towards the goal.

Diagrammatic Representation

Three Dimensions of Perceptual Experience. We view the experience corresponding to visually perceiving a scene as having three aspects or stages. The first and most basic dimension is that of seeing the world as composed of objects – the figure, a set of objects, is discriminated from ground. With diagrams, this dimension corresponds to perceiving it as composed of objects – or figures – as 2-d shapes. The second dimension is that of *seeing as*, e.g., recognizing an object as a telephone, or a figure in a diagram as a triangle. The recognition vocabulary comes in a spectrum of domain-specificity. The third dimension is *relational* – seeing that an object is to the left of another object, taller than another, is part of another, etc. The result of these latter two dimensions of perception is not the spatiality of the objects as such, but symbols and symbol structures that assert and describe, albeit in a vocabulary using spatial terms.

DRS

DRS is intended to capture the result of the figure-ground separation and seeing the diagram as composed of diagrammatic objects with spatial extent. In our current work we consider diagrams that are "black and white" (i.e., no grey scale carrying domain information), though the framework can be extended to account for additional types of diagrams. While objects in an external diagram of this type always appear as regions, they are intended as configurations of three kinds of objects*, point, curve* and *region*. For example, a city in the US map might appear as a small circle, but the reader takes it as a point; the diameter of the circle might symbolize a label for the size, but it is not intended to be proportional to the area of the city. DRS is a representation of the abstract diagram intended. Again, a physical diagram may have alphanumeric labels to be seen as attached to objects, and they may also have pictorial iconic aspects, such as a picture of a church at its location to signal the type of building. DRS does not contain the spatial information corresponding to these various symbols, but uses this information as symbols attached to the objects. Just as a symbol in a symbolic representation in an external medium is a pointer to an abstract symbol in the agent, a diagram on a piece of paper is a pointer to a more abstract diagram of the sort that is in DRS.

DRS is generic. Information that is domain-specific, e.g., that a certain curve is a defensive perimeter, is simply treated as abstract symbolic information, and incorporated as labels attached to the objects to be interpreted by the problem solver using domain-specific conventions. At the level of DRS, there is no recognition of a curve as a straight line, or a region as a polygon. These are recognitions that will be made further downstream by perceptual routines. Also, how curves and regions are actually represented in a computational realization is not relevant at the level of DRS. For example, in a specific implementation, a curve may in fact be represented as a sequence of line segments (as we do in our implementation), or as a set of pixels in an array-based representation. These are representational details below the level of the abstraction of DRS, in which such an object is still a curve.

Perceptual Routines

Perceptual routines in our architecture operate on the objects in DRS and produce two kinds of information. One is reperceiving the diagram to identify emergent or lost objects (points, curves and regions) as objects are added, deleted or modified. The second is that object properties and inter-object relations of specific kinds between objects (such as object A is inside object B) are computed in response to problem solving goals. A brief list of routines that are currently available is given below.

Emergent Object Recognition Routines. Intersection-points between curve objects, region when a curve closes on itself, new regions when regions intersect, new regions when a curve intersects with a region, extracting distinguished points on a curve (such as end points) or in a region, extracting distinguished segments of a curve (such as those created when two curves intersect), extracting periphery of a region as a closed curve. Removal of objects that no longer exist when something changes are included.

Object Property Extraction Routines. Examples: Length-of(curve) (this will return a number), Straightline(Curve) and Closed(Curve) will return True or False.

Relational Perception Routines. Examples: Inside (I1,I2), Outside, Left-of, Right-of, Top-of, Below, Segment-of (Curve1, Curve2), Subregion-of (Region1, Region2), On (Point,Curve), and Touches(object1, object2), Angle-value(curve AB, curve BC). Subsumption relations are especially important and useful to keep track of as object emerge or vanish.

Abstractions of groups of objects into higher level objects.

Domain-specificity. Perceptions may be domain-specific because they are of interest only in some domains, e.g., "an L-shaped region." They may also be domain-specific in that they combine pure spatial perception with domain-specific, but nonspatial, knowledge. For example, in a military application, a curve representing the motion of a unit towards a region might be interpreted as an attack, but that interpretation involves combining domain-independent spatial perceptions – such as extending the line of motion and noting that it intersects with the region – with non-spatial domain knowledge – such as that the curve represents the motion of a military unit, that the region's identity is as a military target belonging to a side that is the enemy of the unit that is moving, etc. In our current implementation, it is the task of the problem solver to combine appropriately the domain-independent perceptions with domain-specific knowledge to arrive at such conclusions, but in application-dependent implementations of the architecture, some of these perceptions might be added to the set of perceptual routines.

Action Routines

The problem solving process may modify the diagram – create, destroy or modify objects. Typically, the task – the reverse of perception in some sense – involves creating the diagram such that the shapes of the objects in it satisfy a symbolically stated constraint, such as "add a curve from A to B that goes midway between regions R1 and R2," and "modify the object O1 such that point P in it touches point Q in object O2." Constructing or modifying diagrammatic elements that satisfy such constraints involves a set of Action Routines parallel to Perception Routines. Similar to PRs, ARs can vary in generality. An AR that we haven't yet used, but we think would be especially valuable, is one that changes a region object into a point object and vice versa as the resolution level changes in problem solving, such as a city appearing as a point in a national map, while it appears as a region in a state map.

Functional Character of the Representation and the Routines

There is a tendency to assume that diagrammatic reasoning is defined in terms of some kind of array computing, in which a 2-D array represents the pixels of the diagram. However, for us, the essence of diagrammatic reasoning is its functional character; namely, a diagram is a functional specification of the spatial information about diagrammatic objects. The various perceptions and actions are also functionally defined. How they are represented and the corresponding algorithms are not of interest at the level of the diagrammatic reasoning problem solving framework. The framework focuses on the functional roles and the mutual interactions of the diagrammatic and symbolic components. Of course, the computational properties of the representations and algorithms are of interest at the computational theory level of Marr or the Symbol level of

Newell (Newell, 1990), but it is important to keep the levels distinct so the real commitments of the framework are properly grasped.

Experiments in Diagrammatic Problem Solving

We have applied the architecture to a number of problems in Army applications: recognition of maneuvers (Chandrasekaran, 2004), critiquing a proposed route for vulnerability to certain kinds of attacks, and a problem in *entity reidentification*. We describe briefly the last-mentioned experiment to indicate how the architecture actually works.

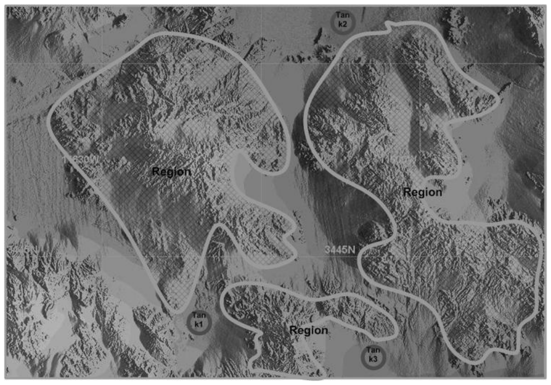
Diagrammatic Reasoning for Entity Reidentification

The problem of entity reidentification arises as a subtask in defense systems that are intended to identify and keep track of enemy assets, based on a variety of information sources, such as satellite photos, human reports, sensor reports and so on. A typical problem arises as follows. A report is received of a sighting of a vehicle of a certain type at a certain location and time. The task is to decide if the new sighting is the same as any of the entities in its database of earlier sightings, or an entirely new entity. Reasoning has to integrate information from different sources – database of sightings, capabilities of vehicles, sensor reports, terrain and map information – to make the decision. Our experiment uses a generic fusion engine[2] for generating, evaluating, combining, and modifying hypotheses about the entity. Diagrammatic reasoning is used to handle spatial aspects: Possible routes, whether routes intersect sensor fields, and how to modify routes to avoid sensor fields.

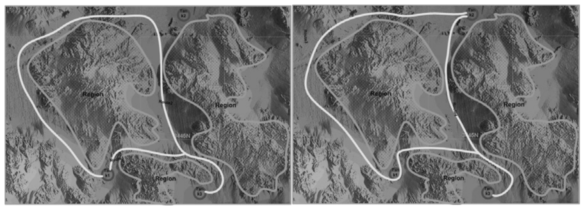
We will refer to the main problem solver as the Fusion Engine or FE in the rest of the discussion.

In the first figure, the terrain map is shown along with three mountainous regions that are drawn to indicate that they are no-go areas for the vehicle types of interest. (The regions are manually drawn, corresponding to a pre-drawn diagram.) The newly sighted vehicle is the small circle at the bottom right. The FE asks the database for previously sighted vehicles of the same type as the newly sighted one, along with the times and locations. The database has responded with two vehicles that are indicated in the diagram below as two small circles, one at the top and the other at bottom left. Let us call them T1 and T2 (with the new vehicle denoted by T3).

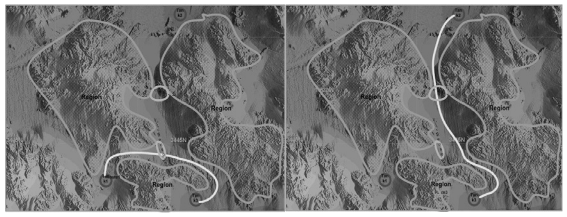
The DRS at the point consists of the three region objects and three point objects. The FE determines the maximum distance T1 and T2 could have traveled given their maximum speed and times elapsed, and asks the diagrammatic reasoner to identify possible paths that the two previous vehicles might have taken to get to the



location of the newly sighted vehicle. The diagrammatic reasoner's Action Routine comes up with two possible paths for each of the vehicles, along with their lengths. The figures below show the routes that each of the vehicles could have taken. The FE discards the longer of the two routes for each of the vehicles as being too long for the time available.



The FE now asks the sensor database for information about sensor fields in the area of interest, including their locations and if they reported any vehicle crossings during the period of interest. It gets back information about the locations of two sensor fields that exist in the region, and that neither of them reported any vehicle crossings. FE asks the diagrammatic engine if the routes for each of the vehicles cross either of the sensor fields. The next figure

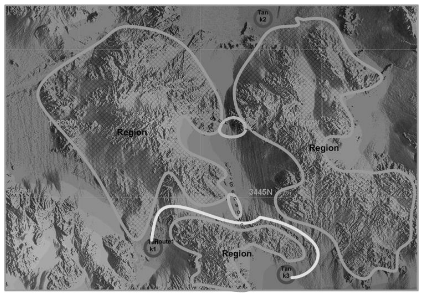


shows the situation. The diagram has been updated with two sensor fields (represented by two small region objects, one at the top and the other closer to the bottom), and we can see that each of the routes indeed crosses one of the sensor fields. The diagrammatic reasoner's perception system extracts the same information and passes it on to the FE.

2 The fusion engine and the problem solving strategy are due to our colleagues John Josephson and Vivek Bharathan.

The routes are not exact paths, but representative members of a class of routes, all of which have the property that they pass between two no-go regions and do not exceed the length constraint. FE now checks if the vehicles routes could have avoided the sensor fields so that the non-reporting by sensor fields could be explained. So it asks the diagrammatic component if the routes could be modified so as to avoid the regions corresponding to sensor fields. The diagrammatic reasoner concludes that the route for T1 could not be modified, but constructs a modified route for T2 as shown in the next figure.

FE now concludes that the most plausible hypothesis is that the newly sighted vehicle is T2 taking the lower route but avoiding the lower sensor field.



The figures that are shown in this discussion are from the display that is updated during the reasoning. The diagrams on the display are generated and updated from the DRS within the diagrammatic reasoning system. All the constructions of diagrammatic elements (such as routes) and perceptions (such as that a route intersects a sensor region) are all computed by the diagrammatic reasoner using DRS and a variety of action and perception routines. The display itself does not play a role in the reasoning, except when a user introduces new diagrammatic objects, which are then used to update DRS.

Related Work

This work is an evolution of earlier general proposals made in (Chandrasekaran, 1997). The following is a sampling of work in diagrammatic research on mixing the symbolic and perceptual modes. Geometry theorem proving has especially been a task that has seen much research in this area, with (Lindsay, 1998) and (Koedinger, 1995) being well-known recent examples. Diagrammatic and symbolic representations in device behavior understanding are considered in (Narayanan, 1995). While there are many points of contact between the representation of the diagram in these systems and DRS, our overall goal is a generic architecture for bimodal problem solving in contrast to the above research whose proposals for primitive objects and perceptions are tailored to the needs of their domains. In contrast, we have little to say about the specific ways in which diagrams work in their domains, but our more general representations may be specialized to their needs. (Ferguson, 2000) describes a program that recognizes domain-specific objects (such as tea kettles) from diagrams composed of straight line segments, and extracts spatial relations between the objects, but does not deal with problem solving in general. Pineda (Pineda, 1989) treats diagrams as sentences constructed from diagrammatic primitives and focuses on the semantics of such sentences. In an informal sense, DRS is such a sentence, with the semantics of the symbols for diagrammatic objects being directly represented by its spatial specification. Heterogeneous representations from a logic perspective are considered in (Barwise, 1995), and in database retrieval in (Anderson, 2002), neither of which is concerned with general issues in problem state representation. In work on general cognitive architectures such as ACT-R (Anderson, 1998) and Soar (Newell, 1990), there has been significant recent emphasis on systems that deal with perceptual input, but the degree of integration of perception that we are proposing has not been attempted in those research areas. The term “perceptual routines” is related to “visual routines” of (Ullman, 1984). We use the former term since our long-term interest in extending the framework to other modes of perception.

It is worth mentioning that the general approach proposed here is here has points of commonality with the proposal in (Barsalou, 1999) on perceptual symbol systems. The elements of the DRS in our architecture are visual symbols and DRS itself is a visual symbol structure in his sense of the terms.

Discussion

Our interest in diagrammatic reasoning is at least partly as a window into cognitive architecture. Our long term goal is to extend the current architectures that focus exclusively on the discrete symbolic component and the corresponding sentential representations to multi-modal states where perceptual and kinesthetic modes have equal status in the production of intelligent behavior. Diagrams provide a nice intermediate point to work – they are practically important in view of their ubiquity in human problem solving activity, and raise the basic issues that a general multi-modal theory of cognitive state will face. Specifically, the issues have to do with how the various modes are represented to retain the essential properties of their mode, and how the multiple modes interact with each other in solving problems. Our proposal for diagrams says that internal representations are already partly interpreted as modal objects, such as points, curves and regions in DR and similar such objects in other domains. This is in contrast to the actual external diagram that is an array of uninterpreted light intensities. We show how in a goal-directed manner the symbolic and diagrammatic components interact with each other: internal perceptual routines may supply relational information in symbolic form to the symbolic component, and the symbolic component may cause the creation and modification of the diagrammatic component.

A richer theory of cognitive state with components from multiple modes can help in a number of ways. It provides at least a partial relief from the computational burden of the Frame Problem. Partly, it is just a matter of using the external world as a constraint enforcer, so that instead of inferring all the consequences of actions, one may just look at the external world, or a perceptually accessible representation of it, and pick up the information. Also, memory is multi-modal as well. Instead of processing all the perceptual components of experience for sentences that may be extracted from it – which are typically too numerous – storing the experience in some perceptual form and applying internal perception as and when needed can be computationally more effective. The diagrammatic reasoning framework shows how that might work in general.

As a technology, the DRS/perceptual routines/action routines component can work with a variety of symbolic problem solving systems. The routines are extensible and DRS is generic. We have experimented with the usability of the technology by trying to apply it to a variety of problems in Army applications. In (Chandrasekaran, 2004), we reported on the use of the technology for maneuver recognition. Recently, we were able to build simple demonstration systems for critiquing courses of actions with respect to vulnerabilities for attack.

Acknowledgments

This paper was prepared through participation in the Advanced Decision Architectures Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0009. Over the years, we have benefited significantly from collaborations and discussions with John Josephson, Robert Winkler, Greg Turner, and Vivek Bharathan.

References

Anderson, M. and B. Andersen, 2002. Heterogeneous Data Querying in a Diagrammatic Information System, in *Diagrammatic Representation and Inference*, M. Hegarty, B. Meyer, and N.H. Narayanan, Editors. Springer: New York. p. 109-111.

Anderson, J.R. and C.J. Lebiere, 1996, *The Atomic Components of Thought*. 1998: Lawrence Erlbaum AssociatesBarwise, J. and J. Etchemendy, Heterogeneous Logic, in *Logical Reasoning with Diagrams*, G. Allwein and J. Barwise, Editors. Oxford University Press: New York. p. 179-200.

Barsalou, L. W., 1999. "Perceptual symbol systems." *Behavioral and Brain Sciences* **22**: 577-660.

Chandrasekaran, B. 1997. Diagrammatic Representation and Reasoning: Some Distinctions. In AAAI Fall 97 Symposium Series, Diagrammatic Reasoning. Boston, MA: American Association for Artificial Intelligence.

Chandrasekaran, B, Unmesh Kurup, Bonny Banerjee, John R. Josephson and Robert Winkler. 2004. An Architecture for Problem Solving with Diagrams, in Diagrammatic Reasoning and Inference, Alan Blackwell, Kim Marriott and Atsushi Shomojima, Editors, Lecture Notes in Artificial Intelligence 2980, Berlin: Springer-Verlag, 2004, pp. 151-165.

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

Koedinger, K.R. and J.R. Anderson, 1990. Abstract Planning and Perceptual Chunks: Elements of expertise in geometry. *Cognitive Science*, **14**: p. 511-550.

Lindsay, R., Imagery and Inference 1995. In *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, J. Glasgow, N.H. Narayanan, and B. Chandrasekaran, Editors., AAAI Press/MIT Press: Menlo Park, CA. p. 111-136.

Narayanan, N.H., M. Suwa, and H. Motoda, 1995. Hypothesizing behaviors from device diagrams, in *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, N. Glasgow, N. H. Narayanan, and B. Chandrasekaran, Editors. The MIT Press: Cambridge, MA. p. 501-534.

Newell, A., 1990. *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Pineda, L., 1989. Graflog: a Theory of Semantics for Graphics with Applications to Human-Computer Interaction and CAD Systems, Ph. D. Thesis, University of Edinburgh: Edinburgh, UK. p. 197+vii.

Shimojima, A., 1996. Operational Constraints in Diagrammatic Reasoning, in *Logical Reasoning with Diagrams*, G. Allwein and J. Barwise, Editors, Oxford University Press: New York. p. 27-48.

Ullman, S., 1984. Visual routines*.* *Cognition*, **18**: p. 97-159.