DESIGN AND USE OF A BIMODAL COGNITIVE ARCHITECTURE
FOR DIAGRAMMATIC REASONING AND COGNITIVE MODELING

DISSERTATION
Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of the Ohio State University

By
Unmesh Kurup

The Ohio State University
2008

Dissertation Committee:
Dr. Balakrishnan Chandrasekaran, Advisor
Dr. John R. Josephson
Dr. James T. Todd

Approved by
Graduate Program in Computer Science
and Engineering
ABSTRACT

In 1948, Tolman introduced the term *cognitive map* to refer to the representation of large-scale space in rats that allow them to remember and navigate back to food sources in a maze. Since then there have been a variety of studies that have sought to explore the nature of the cognitive map, both in humans and in other animals. While these studies have yet to conclusively establish the structure and processes involved, they have been invaluable in identifying characteristics of the representation as well as phenomena exhibited by agents engaged in spatial problem solving. Together, these characteristics and phenomena provide constraints on the nature of the cognitive map and on the processes that are involved in its learning and use. Over the years, there have also been a number of computational proposals for the representation of and reasoning about large-scale space. Some proposals, such as the Spatial Semantic Hierarchy, try to provide broad coverage of the issues from how representations are learned while navigating the world to their subsequent use in problem solving, while others such as those by Bakowsky or Yeap concentrate on explaining certain aspects or phenomena. The problem with these proposals is that the end product is a theory that is self-contained with its own set of representations and processes, whereas in reality, the cognitive map and associated processes are part of the larger cognitive apparatus. Thus, while spatial representations and processes may have unique aspects, it is likely that they also share underlying
representations and mechanisms with those involved in general problem solving. It is important then, both for reasons of accuracy in cognitive modeling and economy in agent building, that spatial representation and reasoning systems be well integrated with general-purpose problem solvers. However, given the differences in mechanisms, assumptions and constraints among these systems, integrating them is a challenging task.

An ongoing area of research in AI and cognitive science has been the unification of cognitive theories within the framework of a cognitive architecture. In this methodology, the behavior exhibited by an agent can be explained in terms of the agent’s architecture, its knowledge and the environment it is in. The architecture is the underlying infrastructure of the agent and can be thought of as those parts that remain unchanged (or change very slowly) over significant periods of time and over application domains. The content, in essence its knowledge, changes often, within and across applications and domains. Thinking of cognition as emerging from a combination of the architecture and the content has multiple advantages both for cognitive science and for AI. For cognitive science, the architecture represents that which is common between a certain class of agents and the differences between them can then be explained in terms of their content (knowledge, goals etc). Moreover, phenomena exhibited by agents can often be explained in terms of one underlying model (the architecture), rather than a multitude of task-specific models. Besides, the constraints provided by the architecture prevent the modeler from making generous and/or false assumptions about architectural constraints during the modeling process. For AI, an architecture provides a set of constraints (memory, learning mechanism etc) that interestingly limit how different agent behaviors are realized. It is also a common platform which can be reused in creating a variety of agents. Lastly, a common architecture makes it easier for models built within the architecture to be brought together to create a unified agent.

This work proposes the use of the cognitive architecture methodology to investigate issues in the representation of and reasoning about large-scale space. In particular, we use biSoar, a version of the cognitive architecture Soar that we have augmented with the Diagrammatic Representation System (DRS). Soar is a well known cognitive architecture for constructing general cognitive systems that perform a wide
variety of tasks. DRS is a system that represents diagrams as a collection of diagrammatic objects each with their complete spatiality. DRS also provides a set of perceptual routines for extracting relational information from the diagram and a set of action routines for constructing and modifying diagrams according to given constraints. In biSoar, information can thus be represented both symbolically and diagrammatically, as appropriate. We describe the design and implementation of biSoar and explain the advantages of this bimodal architecture in problem solving and cognitive modeling about space.

In problem solving, the additional modality allows for more efficient reasoning. We highlight this through the use of examples in the blocks world domain. These examples also serve to describe the bimodal problem solving process. The additional diagrammatic representation also has implications for a solution to certain aspects of the Frame Problem.

To qualify for the task of modeling spatial representation and reasoning, biSoar must exhibit a set of characteristics – represent information in chunks or pieces (both symbolic and metric), intelligently combine these chunks during the problem solving process, learn from problem solving instances and show transfer of learning within and between related but different tasks. We investigate each of the above characteristics, and through the use of wayfinding examples show how biSoar exhibits all of them. We then demonstrate biSoar’s usefulness and its flexibility and versatility in modeling spatial phenomena using three examples. In the first example, we show how biSoar can model a common spatial recall phenomenon called *simplification* as arising out of the architecture rather than as a result of the use of specific strategies or particular pieces of knowledge. In the second example, we show how distortions in the recall of geographic information that are commonly found in studies of human spatial performance can be modeled using biSoar. In the third example, we show how biSoar can model the influences of goals on the representation in the cognitive map. In the case of first two examples, we provide alternative biSoar models that offer different explanations for each phenomenon, demonstrating biSoar’s flexibility.
Dedicated to my parents:
for their sacrifices, both big and small, that allowed me to follow my dreams
I wish to thank my advisor, Chandra, for his intellectual support, patience, encouragement, and most importantly, his inspirational passion for research and his never-ending curiosity about the way the world works.

I thank John Josephson for the valuable ideas and comments that he has provided over the course of many years as well as for his service on my committee.

I am grateful to James Todd for serving on both my candidacy and dissertation committees.

I am indebted to my girlfriend, Erin, whose understanding, encouragement and support was invaluable in keeping me focused on my research.

I would also like to thank my colleagues Vivek Bharathan and Bonny Banerjee for the many stimulating conversations, the fun road trips and for making the lab an enjoyable place to work.

I am also thankful to Tamera and others in the department’s administrative group for their encouragement, support and their invaluable help in shielding the students from bureaucracy and allowing us to concentrate on our research.

Finally, I am grateful for to the Advanced Decision Architectures Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0009 for supporting this work.
VITA

July 17, 1978..................................... Born - Kochi, India
2000................................................... B. Tech, Computer Science and Engineering,
  Cochin University of Science and Technology
2000 - present.................................... Graduate Teaching and Research Associate,
  The Ohio State University

PUBLICATIONS

Research Publications

1. U. Kurup, B. Chandrasekaran, “Modeling Memories of Large-Scale Space Using a
   Bimodal Cognitive Architecture.” 8th International Conference on Cognitive

   Architectural Explanation of Spatial Reasoning, AAAI Spring Symposium on Control
   Mechanisms for Spatial Knowledge Processing in Cognitive / Intelligent Systems,
   Stanford University, California, 2007.

   solution to the frame problem.” Paper presented at the 28th Annual Conference of the


FIELDS OF STUDY

Major Field: Artificial Intelligence

Minor Field(s): Cognitive Science, Software Engineering
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iv</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>v</td>
</tr>
<tr>
<td>Vita</td>
<td>vi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xiv</td>
</tr>
<tr>
<td>Chapters:</td>
<td></td>
</tr>
<tr>
<td>1  INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2  BISOAR: A BIMODAL COGNITIVE ARCHITECTURE</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Multi-modal Cognitive Architectures</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Cognition, Symbol Systems and Multi-modality</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Cognitive Architectures</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Diagrams and Diagrammatic Reasoning</td>
<td>13</td>
</tr>
</tbody>
</table>

ix
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Different types of representations possible in representational theories</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Traditional and multi-modal views of cognition</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Diagrammatic Object Class Hierarchy</td>
<td>20</td>
</tr>
<tr>
<td>2.4 The Diagram Class Structure</td>
<td>21</td>
</tr>
<tr>
<td>2.5 Examples of operator proposal and operator application rules in Soar</td>
<td>24</td>
</tr>
<tr>
<td>2.6 (a) Blocks World and (b) Soar’s WM representation of the world in (a)</td>
<td>25</td>
</tr>
<tr>
<td>2.7 biSoar representation of the world shown in 2(a)</td>
<td>26</td>
</tr>
<tr>
<td>2.8 Examples of operator proposal and application rules in Soar</td>
<td>27</td>
</tr>
<tr>
<td>3.1 (a) Simple BW example (b) Initial contents of Soar’s WM (c) Soar’s WM after operator proposal (d) Soar’s WM after Move applied</td>
<td>43</td>
</tr>
<tr>
<td>3.2: (a) Initial contents of biSoar’s WM (b) biSoar’s WM after Move applied</td>
<td>44</td>
</tr>
<tr>
<td>3.3 Initial state for the blocks world example</td>
<td>45</td>
</tr>
<tr>
<td>3.4 The problem solving sequences for standard and multi-modal Soar for the problem in Figure 3.3</td>
<td>46</td>
</tr>
<tr>
<td>4.1 A high-level overview of the components of a complete agent</td>
<td>55</td>
</tr>
<tr>
<td>4.2 An overview of a complete biSoar agent</td>
<td>56</td>
</tr>
</tbody>
</table>
4.3 A Curve and the effect of detail loss on the curve ............................................ 57
4.4 An example where the agent needs to recall the exact spatiality of curves ..... 58
4.5 The effect of location inexactness and detail loss on (a) point (b) curve and (c) region objects. Original objects are black and recalled objects are red .......... 61
4.6 A high-level view of the current implementation of a complete biSoar agent 62
5.1 Examples of biSoar rules used in the wayfinding task .............................. 71
5.2 A map showing some of the main routes in Columbus ............................ 72
5.3 Information in agent’s LTM for the wayfinding task ............................... 74
5.4 The wayfinding strategy used by the biSoar agent ................................. 75
5.5 Route found by the biSoar agent from P4 to P2 ........................................ 75
5.6 Knowledge learned by the agent in solving task 1 ............................... 77
5.7 Knowledge learned by agent in solving task 2 ............................... 78
5.8 Map created by the agent for the geographic recall task .......................... 78
5.9 Path found by the agent from R1R4 to P2 in Task 3 ............................... 79
6.1 Routes found by Simp1 from the map in Fig 4.2(a). (a) R1R4 to R1R5. (b) P4 to R1R3-1. (c) P1 to P2 ................................................................. 86
6.2 Routes found by Simp2 from the map in Fig 4.2(a). (a) R1R4 to R1R5. (b) P4 to R1R3-1. (c) P1 to P2 ................................................................. 87
6.3 A hierarchical representation of knowledge in the SD-Reno example ....... 89
6.4 The hierarchical LTM representation of the SD-Reno example in MIRAGE. .............................................................................................. 91
6.5 Visualization by MIRAGE in the SD-Reno example................................. 92
6.6 (a) External representation (b) Simplified map (c) Simplified California region (d) Simplified Nevada region (e) Detailed pieces of California region ..... 93
6.7 (a) Map of SW U.S. in LTM (& WM) of Model 1. (b) & (c) are diagrams in WM constructed by Models 2 & 3 ..................................................... 94
6.8 A map showing some of the main routes in Columbus ........................ 96
6.9 list of locations in the route goal ............................................................... 97
6.10 list of adjoining locations in the survey goal........................................... 97
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Space requirements for Soar and biSoar agents</td>
<td>49</td>
</tr>
<tr>
<td>3.2 Time requirements for Soar &amp; biSoar agents</td>
<td>50</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

In 1948, Tolman (Tolman, 1948) introduced the term *cognitive map* to refer to the representation of large-scale space in rats that allows them to remember and navigate back to food sources in a maze. Though he referred to it as a map only in the functional sense, early speculation revolved around whether this representation was really a map (a global metrical representation) or merely a collection of representations (only a minority of which were metrical). McNamara (T. P McNamara, 1986) presents an overview of the debate. Since then, a substantial body of literature has grown around representation and use of knowledge of large-scale space, from a variety of fields including Psychology, Cognitive Science, Artificial Intelligence, Robotics and Geographical Information Systems (GIS). Each group has studied the problem from a different perspective - Psychologists/Cognitive Scientists from a desire to understand and account for human performance in navigation, memory for routes and related spatial tasks; Computer Scientists/Roboticists to provide robots with suitable tools for representing the world and solving problems (such as navigation) within it; and Geographers in the hope that an understanding of how humans represent space will lead to the building of better mapping tools that allow for easier user interpretation and interaction. The problem of human
large-scale space cognition is a significant part of the overall cognition problem and involves perception and motor action as well as cognitive activities such as problem solving and learning. Possibly in response to the enormity of the task, researchers have focused on investigating and explaining smaller sub-parts of the overall spatial reasoning problem. In addition, the differences in goals and methodologies of the disciplines can be observed in the kind of explanations that are often provided for the various spatial phenomena. One such effect that is of particular interest to this thesis is the lack of computational proposals for human spatial cognition. Roboticist proposals are rarely created with the goal of psychological validity or explanatory power with regard to human spatial cognition, while psychological models are rarely computational and often confined to explaining certain aspects or phenomena exhibited by human spatial cognition. There are exceptions of course, most notably Kuipers Spatial Semantic Hierarchy (SSH) (Kuipers, 2000), that try to bridge this gap, but such proposals are few and far between.

Even in the rare cases where the proposal is both computational and a model of human spatial cognition, there is still a question of how well it fits into the overall picture of human cognition, a problem that bedevils most modeling work. In part to counteract this failing, Newell (A. Newell, 1990) proposed a more holistic approach to modeling and agent creation using the cognitive architecture approach. In this approach, the behavior exhibited by an agent is explained in terms of the agent’s architecture, its knowledge and the environment in which it acts. The architecture is the underlying infrastructure of the agent and can be thought of as those parts that remain unchanged (or change very slowly) over significant periods of time and over application domains. The content, in essence the agent’s knowledge, changes often, within and across applications and domains. Thinking of cognition as emerging from a combination of the architecture and the content has multiple advantages, both for cognitive science and for AI. For cognitive science, the architecture represents that which is common between a certain class of agents, and the differences between them can then be explained in terms of their content (knowledge, goals, etc). Moreover, phenomena exhibited by agents can often be explained in terms of one underlying model (the architecture), rather than by a multitude of task-specific
Besides, task-specific models generally do not take into account constraints that a general cognitive architecture has to satisfy and, as a result, such models implicitly assume generous or false assumptions. For AI, this approach provides a common platform which can be reused in creating a variety of agents and makes it easier for models built within the architecture to be brought together to create an unified agent. It also provides a set of constraints (memory, learning mechanism, etc) that interestingly limit how different agent behaviors are realized.

Architectural proposals can be varied. One set of proposals divide the architecture into two parts – a high-level or central cognition module that is responsible for deliberative goal-oriented behavior interacting with modules that perform perceptual and motor functions. These proposals focus on the nature and design of central cognition while leaving the specifics of the other modules to perceptual and motor theorists. The two dominant proposals that share this outlook, Soar (A. Newell, 1990) and ACT-R (J. R. Anderson & Lebiere, 1998), also share a commitment to cognition as rule-based symbolic activity. In Soar/ACT-R, the knowledge that an agent has, the current state of the world and its set of goals are all maintained as symbol structures, where the symbol structures are “linguistic” in nature, i.e., they are similar to the logical forms of natural language sentences. These architectures can be taken as realizations of the “Language of Thought” hypothesis. The underlying ontology is that of a domain of discourse where there are individuals between which various relations hold. All cognitive behavior results from application of inference rules of various sorts to representations in this framework. Since the relationships and individuals can be abstract, the rules can enable inferences by linking information that is far removed from any specific perceptual modality. The generality and potential for linking distal items of knowledge has been a powerful feature of such proposals, supporting open-ended reasoning. However, this commitment is by no means universal in AI and Cognitive Science.

Chandrasekaran (B. Chandrasekaran, 2002a) has proposed that these architectures do not account for the role played in cognition by perceptual modality-specific representations and operations. While the role of mental images in reasoning and problem-solving has been controversial, there is overwhelming evidence (Stephen M.
Kosslyn, 1994; Stevens & Coupe, 1978) (R. Lindsay, 1995) (Johnson-Laird, 1983) that agents have internal experiences in different perceptual modalities, and at times perceptual operations are applied to such representations to obtain information during cognitive behavior. For example, when told that “A is to the left of B and B to the left of C” and asked the question “Is A to the left of C?”, most people report imagining a diagram and extracting the answer from it. We believe that the time is ripe to construct and experiment with multi-modal cognitive architectures and that the modeling of large-scale space offers a suitable domain in which to exercise the abilities of such an architecture. To this end, our research strategy has been to augment the predicate-symbolic representation of Soar with representations and processes that correspond to other modalities. In particular, given the predominance of diagrams and such visual representations in reasoning, an initial attempt should focus on adding the diagrammatic aspects of visual modality. The ability to use such modality-specific representations has multiple advantages: – one, modality-specific computations can be performed independent of each other. This parallelism, when it can be exploited, can help speed up processing. Two, certain types of problems are much more easily solved using a modality-specific representation and accompanying inference mechanisms than through inferences on predicate symbolic representations. Three, having such representations increases our understanding of concepts which have some characteristic properties that are expressed in that modality. For example, our understanding of an apple is enhanced when we have an image of an apple inside our head.

This thesis is about the design of a bimodal cognitive architecture and the application of this architecture to the problem of the representation of and reasoning about large-scale space. In particular, we describe biSoar, a version of the cognitive architecture Soar that we have augmented with an existing diagrammatic reasoning system called DRS (B. Chandrasekaran et al., 2004), and show how it can be used to build models of large-scale space reasoning. DRS represents diagrams as a collection of diagrammatic objects, each with their complete spatiality. DRS also provides a set of perceptual routines for extracting relational information from the diagram and a set of action routines for constructing and modifying diagrams according to given constraints.
biSoar is thus a bimodal cognitive architecture in which information can be represented both symbolically and diagrammatically, as appropriate.

We describe the bimodal problem solving process in biSoar through the use of examples from the blocks world domain. These examples also serve to exhibit the advantages that a bimodal architecture in reasoning about space. The bimodal representation supported by biSoar also has implications for a partial solution to the Frame Problem.

To qualify for the task of modeling spatial representation and reasoning, biSoar must exhibit a set of characteristics – represent information in chunks or pieces (both symbolic and metric), intelligently combine these chunks during the problem solving process, learn from problem solving instances and show transfer of learning within and between related but different tasks. We investigate each of the above characteristics, and through the use of wayfinding examples show how biSoar exhibits all of them. We then demonstrate biSoar’s usefulness and its flexibility and versatility in modeling spatial phenomena using three examples. In the first example, we show how biSoar can model a common spatial recall phenomenon called simplification as arising out of the architecture rather than as a result of the use of specific strategies or particular pieces of knowledge. In the second example, we show how distortions in the recall of geographic information that are commonly found in studies of human spatial performance can be modeled using biSoar. In the third example, we show how biSoar can model the influences of goals on the representation in the cognitive map. In the case of first two examples, we provide alternative biSoar models that offer different explanations for each phenomenon, demonstrating biSoar’s flexibility.
CHAPTER 2

BISOAR: A BIMODAL COGNITIVE ARCHITECTURE

2.1 Multi-modal Cognitive Architectures

A theory of cognition is an account of the underlying principles of intelligent behavior. Traditionally, theories of cognition have also been representational theories - where the agent is presumed to be capable of holding, recalling and manipulating various states that stand for a belief or concept, or an entity in the external world. Some, such as Brooks (Brooks, 1999), have disputed this claim and have instead proposed non-representational theories with varying degrees of success. We restrict ourselves to representational theories and, in particular to, to those based on symbols and symbol processing systems. Representations in a symbol processing framework are symbol structures composed of relational predicates about individuals in the domain of interest. The state of the agent at any moment – often called ‘cognitive state’ – is symbolic, or more accurately, “predicate-symbolic”, as just described. Cognitive state changes, i.e., thinking, proceeds by the application of inferences to the current state.

The role of perception in such systems is to translate input from the external world into these symbols structures. Similarly, motor action translates symbol structures into movements in the world. This view of cognition is related to the “Language of
Thought” hypothesis that cognition is a product of representations and processes that are language-like.

Research in mental imagery has shown that perceptual mechanisms may have a role to play in cognition in addition to giving information about the external world. The phenomenon of mental imagery suggests that the cognitive state has perceptual and kinesthetic components, in addition to the language-like component. That is, it is multi-modal. This view has recently been elaborated as a research program by Chandrasekaran (B. Chandrasekaran, 2006) (B. Chandrasekaran et al., 2004) and has many points of contact with the Perceptual Symbol Systems proposal by Barsalou (Barsalou, 1999). Chandrasekaran proposes that cognitive state changes may be effected by perception-like
operations and by “simulations” in the various modalities, in addition to inference operations on predicate-symbolic structures. Thus, thinking is a goal-directed sequence of multi-modal state changes. Diagrammatic representations provide an especially good illustrative and practically useful example. Chandrasekaran et al (B. Chandrasekaran et al., 2004) have built an architecture for diagrammatic reasoning. Techniques for representing diagrams, for performing perceptions and creating and modifying diagrams and integrating diagrammatic and symbolic operations have been developed.

While research on the role of imagery has been actively pursued in cognitive psychology, and interest in diagrammatic representations has been growing both in psychology and A. I., research on general cognitive architectures has, so far, not looked into the implications of multi-modal cognitive states. The current thesis investigates this question by building a bi-modal version of the cognitive architecture Soar, and applying it to the domain of reasoning about large-scale space.

2.2 Cognition, Symbol Systems and Multi-modality

As mentioned earlier, the traditional view of cognition is that it is a result of processes that act on representations. The predominant view is that these representations are in the form of symbol structures composed of relational predicates about individuals in the domain of interest and the processes are a set of inference rules that modify these symbol structures. Newell & Simon’s Physical Symbol System Hypothesis (Allen Newell & Simon, 1976) and Fodor’s Language of Thought Hypothesis (Fodor, 1975) both fall within this framework.

Over time, various alternatives, such as connectionist architectures (Fodor & Pylyshyn, 1988) and dynamical systems (Van Gelder, 1995), have been proposed that purport to use neither symbols structures nor representations, but as Chandrasekaran (B. Chandrasekaran et al., 1988) has argued these alternatives may still be viewed as carrying content, albeit in media that are different from the symbol structures described above. Also, from a Knowledge Level perspective, even allegedly non-representational theories can be attributed representational content. Regardless, our concern is only with representational theories and hence, for the purposes of this proposal, we concern
ourselves with only symbol processing systems. Fig 2.1 shows a hierarchical representation of the various distinctions involved the kind of representations in representational theories of cognition.

Symbols have two advantages that make them desirable from the point of view of an artificial cognitive system - composability and their ability to provide distal access. Composability ensures that we can use a limited number of symbols to create an unlimited number of combinations that can in turn represent an unlimited number of facts or situations. Symbols are atomic in this view and can be combined, using syntactic rules, to form non-atomic representational entities called symbol structures. These symbol structures can be further combined with each other and with other symbols to create an infinite number of such structures. This ability of infinite representation through a finite vocabulary, referred to as productivity, is reflected both in human language and in human thought. In language, we are able to use a finite number of words to create an infinite number of sentences. Similarly, we are able to use our brain which is limited in size to think of (and consequently represent) an infinite number of thoughts.

The compositional nature of symbol structures also means that the constituent parts of a symbol structure are themselves either symbol structures or atomic symbols. For example, in the symbol structure $\text{Left-of}(\text{block1, block2})$, the parts $\text{Left-of}$, $\text{block1}$ and $\text{block2}$ are symbols that correspond to entities and relationships in the blocks world domain. A cognitive architecture is able to take advantage of this structural regularity and retrieve additional information about the constituents from memory. Further, since the semantic content of a symbol structure is directly dependent on its syntactic structure, the elements retrieved are relevant to the symbol structure and to the task at hand. As an example, let us consider the structure $\text{Left-of} (\text{block1, block2})$. The retrieval component of the cognitive architecture looks at the constituents, $\text{Left-of}$, $\text{block1}$ and $\text{block2}$, and retrieves everything related to $\text{Left-of}$, $\text{block1}$ and $\text{block2}$ from memory. Assume that looking for $\text{block2}$ results in the retrieval of $\text{Left-of} (\text{block2, block3})$ and looking for $\text{Left-of}$ results in the retrieval of $\text{For all } x,y,z \text{ Left-of}(x,y) \& \text{Left-of}(y,z) \rightarrow \text{Left-of}(x,z)$. The system can now use the retrieved information to infer $\text{Left-of} (\text{block1, block3})$. Thus symbol structures facilitate distal access.
Images, in their normal meaning, i.e. as a configuration of image intensities on some surface, are not composable. An agent might see parts of an object in an image, but the image itself is produced not by composition, but by the process of light reflection from a real world light. Are images composable in the same sense that we just described? Let us first consider a photograph. While we, who look at a photograph, may see parts, such as recognizing noses and ears in the photo of a face, the photograph itself is not produced by composing images of the parts. Rather it is produced by light reflecting off a physical object in the world. A camera doesn't produce the image by composing the images of the parts. For such reasons, images are often contrasted with linguistic symbol structures.

Artificial images on the other hand may be produced by composition. For example, police departments often give a victim an opportunity to describe an attacker by composing, and adjusting the parameters of, parts of a face. On one hand, the constituents of the composition, the images of the nose, the ear, etc., are variable symbols, i.e., it has parameters whose values are adjusted in a continuous way, unlike the symbols that participate in linguistic symbol structures. On the other hand, the human agent is composing them, not always as effortlessly as is the case with linguistic symbols. In fact, humans can create mental images by composing image fragments. Building on such intuitions, Chandrasekaran (B. Chandrasekaran et al., 2004) & Barsalou (Barsalou, 1999) propose composable perceptual symbols. As we will describe later in the proposal, while perceptual symbols have spatial extent like images do, they also have the advantages of compositionality and structural decomposability.

2.3 Cognitive Architectures
The behavior exhibited by any agent can be explained in terms of the agent’s architecture, its knowledge and the environment it is in. The architecture is the underlying infrastructure of the agent and can be thought of as those parts that remain unchanged (or change very slowly) over significant periods of time and over application domains. The content, in essence it’s knowledge, changes often, within and across
applications and domains. Cognitive agents span a range from those that have minimal commitment to an architecture (though they may be implemented on some architecture, they need not share its constraints) and nearly infinite flexibility to those that are completely specified by the architecture and with a minimum of flexibility.

Newell (A. Newell, 1990) explains that viewing cognition as emerging from a combination of the architecture and the content has multiple advantages. For one, the architecture represents that which is common between a certain class of agents and the differences between them can then be explained in terms of their content (knowledge, goals etc). In addition, having an architecture also provides a set of constraints (memory, learning mechanism etc) that interestingly limit how different agent behaviors are realized. Also, the constraints provided by the architecture prevent the modeler from making generous and/or false assumptions about architectural constraints during the modeling process. And lastly, a common architecture allows models built within the architecture to be brought together to create a unified agent.

Figure 2.2: Traditional & Multi-modal views of Cognition
A cognitive architecture can be thought of as having the following attributes (though they may be implemented differently and to varying degrees across different architectures).

- **Memory** – Any agent capable of accomplishing rudimentary tasks needs to be able to store goals, knowledge about the problem domain and possibly even previous experiences. At the very least, the architecture should provide a way to store and retrieve these elements when required. Additionally, a particular architectural design can have consequences such as establishing a grain size for memory as well as time to perform stores and retrievals.

- **Goal Oriented Problem Solving** – One of the critical abilities to achieving real time cognition is to be able to bring to bear only knowledge that is relevant to the task at hand. The ability to have goals allows an agent to achieve precisely this, by retrieving only knowledge that is relevant to the current goal. Architecturally, this calls for the agent to have the ability to have a current goal focus and for its control system to retrieve from memory only knowledge that is relevant to that particular goal.

- **Learning** – Any general cognitive agent should be able to behave flexibly in a dynamic world and since it is not possible to predict all possible situations that an agent may encounter, it needs to be able to learn what to do in a novel situation. Additionally, learning also occurs as a side-effect of problem-solving. This, learning through experience, is a process that is automatic rather than goal driven and since it is independent of the content, the agent architecture should support it.

- **Interaction with External World** – If an agent is to achieve an objective in the real world, it needs to be able to interact with it. The architecture needs to provide a mechanism that allows the agent to perform this interaction.

There are several cognitive architectures currently available in the research community, Soar (Laird et al., 1987), ACT-R (J. R. Anderson & Lebiere, 1998), EPIC (Kieras & Meyer, 1994) and CLARION (Sun, 2006). We concentrate on Soar, which, along with ACT-R, is one of the more widely-used cognitive architectures today. Both Soar and ACT-R have been around for over two decades, have numerous models (the content or programming of an agent for a task is called a model) and have strong and vibrant
research communities associated with them. While both can be used to model human cognition and have much in common, there are also significant differences between them. Soar’s goal is a functional account of cognition. Accordingly, it only captures those phenomena that are involved in intelligence, as abstractly conceived in Newell’s theorizing. Act-R has the added constraints that its architecture has to have a mapping on to the neural framework of the brain and that it exhibit a range of phenomena exhibited by humans. For example, humans forget, but since there have been no reliable research proving that forgetting is a functional requirement for intelligence, Soar’s architecture does not account for forgetting. Act-R on the other hand does have a mechanism that accounts for forgetting. The underlying philosophy in Soar is that the architecture should stand with the fewest mechanisms, see how much explanatory power they provide and then only add further mechanisms as needed. This top level difference between these architectures permeates down to some of the architectural choices that they have made. There are other differences too, but they each have their strengths and weaknesses. Ultimately, the only way to judge whether an architecture is a successful one is by trying to create models for it over a long period of time. If the architecture is capable of handling these various modeling tasks without any significant changes, then there is a good chance that the architecture is sound. The Soar and Act-R communities are committed to this empirical test.

biSoar is built on a framework for diagrammatic reasoning constructed by Chandrasekaran and others (B. Chandrasekaran et al., 2004). Before presenting biSoar, I describe this framework.

2.4 Diagrams and Diagrammatic Reasoning

External diagrams are ubiquitous in everyday problem solving, whether it is to draw a map for navigation, solve an analytical problem or simply as an external memory aid. For example consider the following statements:

1. Point A is to the left of point B.
2. Point B is to the left of point C.
3. Is point A to the left of point C?
If we were to create a diagram from statements 1 and 2, the answer to the question in 3 can be read directly from the diagram. To see why this is powerful, consider solving the same problem using the predicate-symbolic approach. First, statements 1 and 2 are encoded into predicate-logic representations as shown below.

4. left-of(A,B)
5. left-of(B,C)

Next, the system has encoded within it the knowledge shown in statement 3.

6. left-of(X,Y) & left-of(Y,Z) \Rightarrow\leftarrow left-of(A,C)

Substituting for X and Y in 3 with A and B, the system comes up with the answer left-of(A,C).

Now imagine that instead of just three points, there are 20. Looking at the diagram to see if point A is to the left of point T still takes the same amount of time (within certain limits), whereas the predicate symbolic system has to apply the rule in 6 nineteen times to come up with the same answer. The power of diagrammatic reasoning arises due to the nature of physical space. When an external diagram is constructed, physical space enforces certain relationships between the entities in the diagram, like in the case of the relationship between points A and C above. Such relationships are now available for pickup by an agent with suitable perceptual capabilities.

Chandrasekaran (B. Chandrasekaran, 2002a) lists a variety of ways in which diagrammatic reasoning may be used in problem solving including

1. Proposition Extraction
2. Deliberative Reasoning with Visually Extracted Propositions and
3. Proposition projection, i.e., visually simulating motion and changes in position of objects in the diagram and perceiving changes between relations

While there is consensus that external diagrams are helpful, both as memory aids and due to the presence of emergent relationships, there is disagreement about the nature of internal diagrams and the role, if any, of such diagrams in reasoning. The critical issue is whether the internal representation of a diagram is homomorph with respect to a
diagram in physical space. For our purposes, the essential nature of the internal representation is secondary as long as it captures the functionality that an external diagram provides, namely that emergent relationships be made available for pickup. In addition, our initial proposal is to build an ideal version of a bimodal reasoning system (which works well with the minimalist approach of Soar). For the first approximation, issues such as the size of diagrammatic memory or the limited abilities of the human perceptual system are not addressed. Instead, it is assumed that such limitations do not exist for the current system. In the next section, I detail such a system called the Diagrammatic Representation System (B. Chandrasekaran et al., 2004).

2.4.1 Diagrammatic Representation System

DRS was originally presented in (B. Chandrasekaran, Josephson, J.R., Banerjee, B., Kurup, U., Winkler, R., 2002c). What follows is largely taken from (B. Chandrasekaran et al., 2004). The DRS consists of a basic set of primitive objects, creation/modification operations called Action Routines (ARs), and information reading capabilities called Percepual Routines (PRs). The diagram is intended to be functionally equivalent to a diagram – external or internal – used by a human agent during problem solving. Defining DRS functionally avoids many of the contentious issues surrounding whether a computational representation is “really” a diagram. Perceptual routines then act on this to recognize object types and relations. Conversely, action routines may construct or modify aspects of the diagram, as we will discuss shortly.

In the DRS view, a Diagram is a pair (I, DDS) where I is the image, defined as a specification – implicit or explicit – of intensity values for points in the relevant regions of 2-D space, and DDS is the Diagram Data Structure, which is a list of labels for the diagrammatic objects in the image; associated with each object label is a specification of the subset of I that corresponds to the object. A diagrammatic object can be one of three types; point, curve, and region. Point objects only have location (i.e., no spatial extent), line objects only have axial specification (i.e., do not have a thickness), and region objects have location and spatial extent. The labels are internal to DDS. External labels such as A, B, etc., are additional features of the objects that may be associated with them.
In keeping with the functional notion, DDS can represent only a diagrammatic instance (not a class), and its spatial specifications must be complete, just as in the case of an external diagram, and unlike in the case of predicate-based representations that can provide partial specification of a situation, or of classes of situations (such as for all). This does not mean that the agent is committed to all the spatial details in the DDS – it is the task of the problem solver to keep track of what aspects are intended to represent what.

2.4.1.1 Perceptual Routines

Ullman (Ullman, 1984) proposed a set of elementary visual routines, such as visual search, texture segregation and contour grouping that might be composed to perform complex visual tasks. The notion of perceptual routines (PRs) is based on a similar notion of composable and extensible primitives, but more oriented to the needs of problem solving with diagrams. Because of the interest in generic objects, aspects of the proposals are intended to be domain-independent. A PR takes specified objects in a diagram as arguments and produces information corresponding to a specific perception. PRs can be categorized into two classes, the first set producing objects with spatial extent as their output, and the second set producing symbolic descriptions. The first includes PRs that identify emergent objects -- points, curves, and regions -- that are created when a configuration of diagrammatic objects is specified or modified, and similarly objects that are lost. The PRs of the first class are domain-independent in that the point, curve, region ontology is completely general. The PRs of the second class produce symbolic descriptions belonging to one of two kinds: (i) specified properties of specified objects (e.g., curve C has length of m units), and (ii) relations between objects (e.g., point A is in region R, curve C1 is a segment of curve C2, object A is to the left of object B, values of the angles made by intersection of curves C1 and C2).

The PRs of the second class come in different degrees of domain specificity. Properties such as length of curve, area of a region, and quantitative and qualitative (right, acute, obtuse, etc.) values of angles made by intersections of curves are very general, as are subsumption relations between objects, such as that curve C1 is a segment
of curve C2. Relations such as Inside(A,B), Touches(A,B), and Left-of(A,B) are also quite general. PRs that recognize that a curve is a straight line, a closed curve is a triangle, etc., are useful for reasoning in Euclidean geometry, along with relations such as Parallel (Line 1, Line2). The PRs of the second class are open-ended in the sense that increasingly domain-specific perceptions may be conceived: e.g., an L-shaped region, Half-way- between(point A, point B, point C). Chandrasekaran (B. Chandrasekaran et al., 2004) lists a number of perceptual routines that have been identified and implemented, including emergent object recognition routines such as intersection identification, property extraction routines such as Length-of(), Closed(), relational perception routines such as Inside-of(), Left-of() etc, and abstraction routines that combine two or more objects to create a new object.

As a diagram is created or modified, and given a repertoire of PRs, when a specific PR should be applied is a practical computational issue. It is the task of the problem solver to decide which PRs to apply depending on its current set of problem solving goals.

2.4.1.2 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. Again similar to PRs, ARs can vary in generality. Deleting named objects that exist in the diagram, and adding objects with given spatial specifications, e.g., Add point at coordinate, Add curve <equation>, etc., are quite straightforward. ARs include translation and rotation of named objects for specified translation and rotation parameters.

More generally, each of the PRs can be reversed and a corresponding AR imagined. For example, corresponding to the PR Inside(R1, R2) is the AR, “Make region
R2 such that Inside(R1,R2).” (assuming region R1 exists); and corresponding to Length(curve C1) is the AR, “Make curve C1 such that Length(C1) < 5 units.” In most such instances, the spatial specification of the object being created or modified is radically under-defined. Depending on the situation, random choices may be made, or certain rules about creating of objects can be followed. However, the problem solver needs to keep track of the fact that the reasoning system is not committed to all the spatial specification details. The sets of routines are open-ended, but routines that are useful across a number of domains are described in (B. Chandrasekaran et al., 2004).

2.4.2 DRS Implementation Issues

Any implementation has to make far more commitments than are part of the theory, for example, to specific data structure representations over others. In the case of DRS, the theoretical issues can be identified at different levels of abstraction. We review some of the dimensions of potential theoretical interest in DRS.

1. The problem solver’s subgoals require information about spatial properties of objects in the diagram or about spatial relations between the objects, or the subgoals require creation or modification of objects so that they satisfy specified constraints. For this level of specification, a diagram can be defined as a data abstraction with specified operations. There is no theoretical commitment beyond the requirement that the spatiality of each object be available for the operations. At this level, DRS is functionally a diagram, i.e., it supports perception and creation/modification activities that a real diagram on a physical surface does. It is of no theoretical interest how the diagrams are represented, e.g., whether as arrays, algebraic equations that describe various curves, or in some other manner. The definition of DRS in (B. Chandrasekaran et al., 2004) and in this thesis is at this level of abstraction. In section 2.4.3, we describe, in the interest of completeness, how DRS is implemented in our system.

2. How diagrams are represented affects the time and space complexity of the operations, i.e., the perceptual and action routines. For both AI and cognitive modeling, such complexity considerations may be of interest. For example, human
performance in a task such as “find the object that is closest to object A such that A will fit inside,” takes longer if the object of interest is farther away from A than when it is closer. An AI application may involve dealing with diagrams with thousands of objects and the speed of certain perceptions may be important. In such cases, how the objects are represented becomes a matter of theoretical interest. Different representations, e.g., curves as a series of line segments, algebraic equations and diagrammatic objects as arrays support perception algorithms with different properties (Banerjee, 2007). As biSoar evolves, especially so that it can help model a larger set of phenomena in spatial cognition, it will be necessary to explore DRS representations that are closer to human cognition.

2.4.3 DRS Implementation Details
There are a number of different ways in which the functional description above can be realized. The particular choices we make are motivated by ease of implementation and to ensure inter-operability with other third-party applications. This implementation is described below.

A Point in DRS is represented using a pair of XY coordinates. A Curve is considered to be a sequence of straight lines and is represented as a sequence of XY coordinates. Each adjoining pair of points in this sequence represents the end-points of a straight line. A Region is represented by the closed curve that forms its perimeter. Like in the case of a curve, a region object is also represented as a sequence of XY coordinates with the last pair being the same as the first.

Along with the above spatiality information, each object structure also contains additional symbolic information such as a unique identifier for each diagrammatic object, as well as optional symbolic information such as a symbol associated with the object.
Diagrammatic Object
Members
Internal ID – Number
External ID – string
Methods
getInternalID()
getExternalID()
setExternalID()

Point Object
Members
double lon, lat
Methods
set(double, double)
double getlon()
double getlat()
int getType()

CurveRegion Object
Members
double spatialExtent
long numPoints
Methods
double** getSpatialExtent()
setSpatialExtent(double **, long)
long getNumPoints()

Curve Object
Methods
int getType()

Region Object
Methods
int getType()

Figure 2.3: Diagrammatic Object Class Hierarchy
Fig 2.3 shows the object hierarchy of the Diagrammatic Object Class. Additional intermediate classes such as CurveRegion are only for implementation reasons and are not part of the theory. The Diagram itself is simply a collection of diagrammatic objects. Like in the diagrammatic object case, each diagram has a unique identifier as well as other optional symbolic information. The Diagram data structure API provides for methods for adding, retrieving and deleting diagrammatic objects from the diagram. Fig 2.4 shows the Diagram class structure.

Perceptual and Action Routines

While the representation described above retains the complete spatiality (to an arbitrary degree of precision) of the objects in a diagram, the perceptual and action routines that are available determine the functionality provided by the DRS system. That is, even though the representation captures geometric information, if the routines provided only allow the extraction of topological or ordinal information, then the functionality provided by the representation is topological or ordinal respectively. A complete list of currently available routines can be found in Banerjee (Banerjee, 2007).
The DRS system just described can be used on conjunction with a predicate-symbolic problem solver to solve a wide variety of problems, with both spatial and non-spatial sub-parts. In certain cases, the use of diagrammatic reasoning can make the inference process much simpler, allowing the agent to extract the required knowledge from the diagram rather than infer it by the application of rules. Diagrammatic reasoning however has some major drawbacks. For instance, DR (using external diagrams at least) is possible only on the assumption that the agent has a sufficiently complicated perceptual apparatus. Perceptual routines are also often slower than task-specific sentential representations and inference mechanisms. These problems are not hard to overcome though. As technology progresses and humanoid robots become possible, agents will have the required perceptual apparatus and these will hopefully be designed to leverage the generic nature of diagrammatic representations and the innate parallelism of diagrammatic operations.

However, it is the over-specificity of constructed diagrams that is its greatest drawback – How do we know what to believe and what not to believe in a diagram? As an example, consider the diagram created by an agent from the statements “Block A is to the left of block B” and “Block B is to the left of block C.” It is necessary that in the diagram that is created, A not only be to the left of B but also a definite distance to its left. Similarly, there is a specific distance between B and C. It is possible for an agent to wrongly conclude from the diagram that A is x units to the left of B; a mistake that humans rarely make. Without the ability to distinguish what can and what cannot be believed in a diagram, an agent cannot be sure of the answer it retrieves by diagrammatic reasoning. Chandrasekaran (B. Chandrasekaran, 2002b) has commented that mappings between problem domains and diagrammatic representations are prevalent in our culture, passed on from generation to generation and accumulated over an individual’s life span. One answer then to this problem is to outfit an agent with a fairly rich set of such mappings that allow it to reason effectively using diagrams. This, combined with its ability to learn, should ensure the agent’s ability to tackle the problem of over-specificity.
From a cognitive modeling perspective, there is debate as to whether diagrams play any role in reasoning or whether they are merely epiphenomenal. Pylyshyn (Zenon W. Pylyshyn, 2002) and Kosslyn (S. M. Kosslyn, 1990) give an overview of the debate as well as arguments for and against the use of mental images in reasoning. While the debate itself is important, for our purposes its resolution is secondary. The lifespan of this debate (over 20 years) is evidence that it is not an easy debate to settle and that there is sufficient evidence on both sides of the coin. The product of our research then can be taken as an example of how an architecture capable of having diagrams and doing diagrammatic reasoning would look like. It is not meant to definitely answer the imagery debate one way or another but to provide an example of a bi-modal architecture were there, in reality, diagrams in the brain. The existence of such an architecture may then be able to shed light on certain important questions within the imagery debate.

2.5 biSoar

2.5.1 Soar

Soar (Laird et al., 1987) is an architecture for constructing general cognitive systems. Towards achieving this goal, Soar provides representations for short and long-term memory, mechanisms for interacting with the external world, a sub-goaling strategy that is independent of the task and domain, and a learning mechanism that allows Soar to learn as a result of success in solving sub-goals. The Soar architecture also provides a rule-based programming language that can be used to program the Soar agent. Soar’s Working Memory (WM) is represented as Identifier, Attribute, Value triplets (Ex: (S1 Object O1) (O1 Color Red)). Long term memory (LTM) in Soar is a collection of rules. Each rule has a condition (if) part that is matched to WM. If a match exists, WM is changed according to actions specified in the action (then) part. There are two kinds of rules – operator proposal and operator application. Proposal rules propose operators. Each operator can be thought of as the next possible step to take in the problem solving process. Application rules apply the actions of the respective operators. Figure 2.5 shows an example of operator proposal and operator application rules.
2.5.2 Cognitive State in Soar

Soar’s representations are predicate-symbolic. The cognitive state in Soar is represented by the contents of Soar’s WM and operator, if any, that has been selected. Figure 2.6(b) shows the Soar’s cognitive state representation of the blocks world example in 2.6(a). During problem solving, Soar goes through a series of 5 phases – input, proposal, decision, apply and output. In the proposal phase, all operators that are relevant to the situation (that match against conditions in WM) are proposed. In the decision phase, an operator is selected and the corresponding application rule is executed in the apply phase.

Soar provides a set of mechanisms for interacting with the outside world through an io link on the top state. The io links has two sub-states with the attributes input-link and output-link respectively. At the beginning of every decision cycle, Soar goes through an input phase where it senses the environment and appropriately updates the values on the input-link. Any changes to be made to the environment are added as attribute-value pairs to the output-link during Soar’s decision phase. The architecture checks for such changes during the output phase of the decision cycle and appropriately updates the environment.
Soar’s learning mechanism is called **chunking**. If Soar becomes stuck (called an **impasse**), it creates a sub-goal to try and resolve the problem. For example, if, during the decision cycle, Soar does not know which operator to select, it creates a sub-goal to try and choose an operator. The sub-goal goes away when the impasse that created it is resolved and the information that caused it to be resolved is used to create a rule called a **chunk**. The next time Soar is faced with the same problem the chunk is executed instead of re-solving the problem.

To create biSoar, Soar is augmented with the Diagrammatic Reasoning System (DRS).

### 2.5.3 Cognitive State in biSoar

The cognitive state in biSoar is bimodal – it has both symbolic and diagrammatic parts. Fig 2.7 shows the bimodal representation of the world depicted in Fig 2.6(a). Working memory is biSoar is represented as a quadruplet, with each Identifier, Attribute, Value triplet augmented with a diagrammatic component. The diagrammatic component is represented using DRS. It represents the visualization of the triplet. Since not all triplets
need to be (or can be) visualized, these components are present only as needed. States represent the current or potential future state of interest in the world and the symbolic and the diagrammatic part may represent related or distinct aspects of the world. However, the diagrammatic representation is “complete” in a way that the symbolic representation is not. For example, from the symbolic representation alone it is not possible to say without further inference whether A is above C. But the same information is available for pick up in the diagram with no extra inference required. This has advantages (for instance in dealing with certain aspects of the Frame Problem) and disadvantages (over-specificity). The implications of the use of DR are however outside the scope of the current paper.

2.5.4 Utilizing the Diagrammatic Representation in biSoar

The DRS is accessed using the input-output links in Soar. To create/modify/query the diagram, a Soar agent creates an identifier with the attribute “routine” on the output-link.
Depending on the routine, various parameters are added to the identifier. During the output phase of the agent, the biSoar architecture picks up the DRS query, executes it and adds the answer to the input-link. This is then available to the agent during the following decision cycle. Figure 2.8 shows an example of the operator proposal and operator application rules in biSoar. Problem solving in this new bimodal Soar proceeds through the application of rules that act on and modify both the symbolic and diagrammatic sections of the working memory.

2.6 Related Work

Related work in this field can be divided into three categories. The first section is a review of the existing theories of bimodal cognition in Soar and Act-R. The second one consists of theories of cognition that challenge aspects of the classicist sentential model and have points of commonality with the multi-modal view. This includes the Theory of Mental Models, Embodied Cognition, Emulation Theory and Perceptual Symbol Systems. The third set of work, more practical in nature, consists of systems that have a diagrammatic reasoning component. Most such systems have multiple representations and, sentential inferences and perceptual operations are interleaved in the reasoning process. In each case, I will briefly describe the similarities and differences between these proposals and the proposal for multi-modality.
2.6.1 Soar & ACT-R Theories of Bimodal Cognition

**Soar & Soar-VI**– Soar is an architecture for constructing general cognitive systems. Towards achieving this goal, Soar provides representations for short and long-term memory, mechanisms for interacting with the external world, a sub-goaling strategy that is independent of the task and domain and a learning mechanism that allows Soar to learn as a result of success in solving sub-goals. Knowledge in Soar is distributed between its productions and the contents of its working memory (WM). The productions represent Soar’s long term memory (LTM), its knowledge about the domain, problem solving techniques, heuristics, general knowledge etc, while WM contains information that is specific to the task at hand including the current state of the world, proposed actions and any other related information.

Soar’s Working Memory (WM) is represented as Identifier, Attribute, Value triplets (Ex: (S1 Object O1) (O1 Color Red)). Long term memory (LTM) in Soar is a collection of rules. Each rule has a condition (*if*) part that is matched to WM. If a match exists WM is changed according to actions specified in the action (*then*) part. There are two kinds of rules – operator proposal and operator application. Proposal rules propose operators. Each operator can be thought of as the next possible step to take in the problem solving process. Application rules apply the actions of the respective operators. During problem solving, Soar goes through a series of 5 phases – input, proposal, decision, apply and output.

In the proposal phase, all operators that are relevant to the situation (that match against conditions in WM) are proposed. In the decision phase, an operator is selected and the corresponding application rule is executed in the apply phase. Soar’s learning mechanism is called *chunking*. If Soar becomes stuck (called an *impasse*), it creates a sub-goal to try and resolve the problem. For example, if, during the decision cycle, Soar does not know which operator to select, it creates a sub-goal to try and choose an operator. The sub-goal goes away when the impasse that created it is resolved and the information that caused to be resolved is used to create a rule called a *chunk*. The next time Soar is faced with the same problem the chunk is executed instead of re-solving the problem. Soar interacts with the external world through a link called the *input-output* link.
in its WM. Since Soar’s WM is made up of attribute-value pairs, input from the external world is selected by an attention mechanism and converted to symbolic attribute-value pairs before being placed on the input link. This action takes place at the beginning of every Soar decision cycle. Similarly, commands to motor systems are transmitted by writing them to the output link from where they are taken at the end of every decision cycle and converted into an appropriate form before being passed on to the motor mechanisms. The attention and conversion mechanisms are both outside the purview of Soar but they can, of course, be controlled by commands using the input-output link.

Using this setup, it is possible for Soar to interact with the external world or a representation such as a diagram. However, internally, Soar has only a single representational system and a single inference mechanism both of which are predicate-symbolic.

Lathrop and Laird (Lathrop & Laird, 2007) have proposed augmenting Soar to include a Visual Imagery component. This new system is called Soar-VI. The imagery component allows Soar-VI to create, manipulate and inspect visual images. The Imagery component has two parts - a depictive representation or image (a 2D array) and a set of operations that can be performed on the representation. The images are diagrammatic, in the sense of the DRS proposal. i.e., an image consists of objects (background subtraction, edge detection etc have been performed) and every object is associated with a collection of array elements in the image. It should be noted that the eventual goal for the Imagery component is to be able to account for all aspects of visual imagery (such as 3D, color etc) rather than just diagrams, but the current implementation is restricted to 2D diagrammatic representations. The Imagery component and DRS share similarities, particularly with respect to the commitment to retaining the spatiality of the depicted objects and providing an interface to create, modify and inspect the representation. The Imagery component however does not propose an ontology for imagery objects, unlike DRS, which has the ontology of points, curves and regions.

The Imagery component is the equivalent of DRS, and when added to Soar forms a multi-modal system. There is one important theoretical distinction however. Our work is based on the assumption that all aspects of the agent’s architecture including the
cognitive state, memory, learning etc, are multi-modal and that during problem solving Soar can seamlessly access representations across all modalities. Diagrammatic representations are, thus, part of high-level cognition. Lathrop and Laird take a different approach, one in which the Imagery component is part of the total cognitive system – accessible by high-level cognition but lying outside of it. This means that perceptual representations can be accessed only through the input-output link and access to them is restricted to the input and output phases of Soar’s decision cycle. It is possible, however, that this is a distinction without a difference. In practice, biSoar and Soar-VI are very similar and their system, like biSoar, can model the visualization of information and subsequent extraction of the desired spatial relationships.

While the Imagery component itself is outside of high-level cognition in Soar-VI, images are stored in Soar-VI’s LTM, specifically, its semantic memory. Semantic memory is a relatively new construct in Soar and is still under development. Soar semantic memory is a declarative style memory oriented towards capturing facts and concepts. One of the aspects of that is still unclear is the process by which elements are learned into semantic memory. Consequently, it is unclear how exactly images are learned into semantic memory in Soar-VI. The learning of diagrammatic elements is not a problem in biSoar because it has only rule-based procedural memory, where learning happens via chunking. The chunking process is extended in biSoar to include the learning of diagrammatic elements (with certain restrictions as detailed in the section describing biSoar).

In conclusion, Soar-VI and biSoar are very similar, with the symbolic representations and processes in each case being identical. The diagrammatic/visual representations have much in common too, though DRS theory is a little better developed. The major difference between the two proposals is how diagrammatic elements are learned and stored into LTM. Soar-VI, which uses semantic memory to store imagery elements, does not have an account of the learning process, while biSoar, which uses procedural memory, does provide an account.
ACT-R & ACT-R/S – ACT-R, like Soar, is an architectural theory of high-level human cognition. While Soar’s goal is to model the functional aspects of intelligent problem solving, ACT-R’s goal is to model all aspects of human cognition – from the functional to the implementation specific (such as forgetting). In ACT-R theory, the architecture consists of a set of modules (goal stack, declarative memory etc) each with a buffer that acts as an interface for that module. A set of procedural rules, whose conditions are based on the contents of one or more of the buffers and whose actions modify one or more of the buffers, form its procedural memory. The declarative memory module forms its semantic memory. ACT-R also has a goal stack which keeps track of the goals of the agent, and perceptual/motor modules that allow the agent to interact with the world. Each module also has a buffer that forms the interface between the module and the LTM rules. The buffers can store one chunk of information, such as a chunk of declarative memory, or a single goal from the goal stack.

Problem solving proceeds in ACT-R by the repeated application of procedural rules until the goal is achieved. At each decision cycle, the ACT-R control mechanism selects the rule whose conditions are satisfied and makes changes according to the specified actions. If there is more than one rule that matches, then it calculates the activation level for each rule and selects the one with the highest activation. ACT-R interacts with the external world by means of a set of modules called the perceptual-motor modules. Perception modules convert information from the world (pixels in the case of visual module, raw audio in the case of the auditory module) into a set of features. These features are combined in different ways, depending on attention, to form a unit of symbolic information called a chunk. Each module also has one or more buffers into which these chunks can be placed for retrieval by ACT-R. Similarly, commands to the motor module are passed on by placing them in the corresponding buffer. As in the case of Soar, the role of the perceptual system in ACT-R is to provide information about the external world in predicate-symbolic form.

The ACT-R architecture is not bimodal. While it has limited facilities for spatial representation (via its perceptual module), these representations are not part of the ACT-R high-level cognitive theory. High-level cognition has no access to these representations
and cannot influence their formation (other than by specifying constraints for attention). Further, ACT-R has no theory of imagery or an explanation for the representations or processes involved in imagination.

ACT-R/S (Harrison & Schunn, 2002) is an expanded ACT-R architecture with mechanisms for object representation that facilitate recognition, manipulation and navigation. This extended ACT-R has three visuo-spatial systems. The visual system, inherited from standard ACT-R, identifies visual features such as edges, does figure-ground separation and recognizes objects based on two-dimensional retinotopic information. The manipulative system has detailed representations of objects, based on Biederman’s geon-based three-dimensional representation, and is used to drive the motor system. The high level of detail in these representations allows the motor system to perform fine-grained operations. The configural system represents the locations of objects in space and is used to aid navigation. The representation is coarse-grained and just detailed enough to allow objects to be avoided during navigation.

The additional representations in ACT-R/S are used solely for the purposes of driving the motor systems (for manipulation) and navigation rather than general purpose spatial reasoning. More importantly, the cognitive process in this augmented ACT-R is still based on the symbolic representations and process pre-existing in the original ACT-R architecture. Information from the new visuo-spatial systems is converted into symbolic chunks before being passed on to ACT-R.

2.6.2 Other Theories of Cognition

There have been multiple alternatives proposed to the traditional “LOT” hypothesis. In general, these theories call for additional representations that are non-sentential. I give a brief overview of these theories and talk about how the multi-modal view of cognition relates to these proposals.

The Mental-Modal Theory (Johnson-Laird, 1983) – Reasoning about the world often involves abstract reasoning such as those modeled by syllogisms. The theory of mental models holds that even in abstract reasoning, people proceed concretely, specifically, by
building models (instances) of the situation and generalizing from them to the solution. Just like models in logic, each mental model represents a possibility and the ability to construct all possible models plays a crucial part in success during problem solving. Reasoning in the mental-model theory involves construction of models and checking for the presence or absence of required relations. This is in contrast to how inference by abstract rule application works. Studies have shown that people take longer to solve problems that require the construction of more models even if the same problem has a much simpler inferential solution (Johnson-Laird, 1998) lending credence to the Mental-Model Theory.

There are important relations between diagrammatic reasoning and mental models. When a diagram is a model of the situation, such as in the case of the Left-of example, reasoning proceeds by checking the general conclusion in the model and generalizing. Structurally, this is similar to the situation in the use of mental models, where also the conclusion is verified in the model and generalized. But at this level of comparison, reasoning in DR and reasoning in Johnson-Laird’s Mental Models are only structurally similar. Additionally, when mental models are used in tasks such as syllogistic reasoning, the models that are built are represented diagrammatically, and the agent “reads-off” the information from the diagram perceptually. This brings another level of commonality between DR and MMs.

**Embodied Cognition** – Embodied Cognition (EC) (M. L. Anderson, 2003) views thinking and reasoning as a real-time activity that is performed by an agent involved in achieving goals in a particular situation existing in the world. In EC, cognition is not just the product of an abstract system, called the mind, involved in manipulating symbol structures, but arises from the interactions between the agent’s brain, its perceptual systems and the particular task and environment it is currently in. Anderson calls this view as one of thinking of humans as “acting beings” rather than as “thinking things.” A consequence of EC’s view is that it allows the agent to use the world as its own model. This frees the agent from having to represent the world and any information about it. Taken to the extreme (Brooks, 1999), agents do not have representations at all. However,
the more common view is that while agents do represent and solve abstract problems, their representations and processes are highly selective, based on eventual purposes and physically grounded.

One way to look at multi-modality is to think of it as bringing abstract representation and reasoning closer to the EC view by introducing modality specific representations and processes. This allows for agents representations and processes to be selective and task-dependent (or at least modality dependent). It could also provide a way for abstract symbols and operations to be physically grounded thereby satisfying EC’s central research goal – the grounding of symbols in terms of the characteristics of the agent’s embodied experience and its physical characteristics.

**The Emulation Theory of Representation** – Emulation theory (Grush, 2004) is based on the idea that the agent doesn’t simply interact with the world; it also constructs anticipatory models of the body and the environment (Visual, Auditory, Motor-action etc). During perception, the agent uses these models in conjunction with the data coming from outside to mediate, enhance and anticipate perception and action. When there is no outside data, as in the case of imagining, these models instead simulate perception leading to the internal experiential aspect of imagination.

Both Emulation Theory and Multi-modality share the idea of internal models that capture the causality of space and allow agents to extract, rather than infer, various relations from the representation. However, models in Emulation Theory are generally geared towards capturing features of an active dynamic world such as the motion of entities through space. While this is the direction that Multi-modality is moving towards, currently it (specifically the DRS) accounts only for reasoning from static diagrams.

**Perceptual Symbol Systems** – Barsalou (Barsalou, 1999) has proposed that cognition is the product of a symbol system whose symbols are underwritten by perceptual elements. In his theory, perception of the external world activates areas in the sensory-motor regions of the brain. Association areas in the brain extract these activations (neural patterns) through the use of selective attention mechanisms and store them. Every activation stored is a conceptually relevant part of the experience (such as a chair or table
or even greenness or hotness) and not simply a holistic image of the entire perception. As more memories of a component are formed, they are organized around a common frame and the brain implements a simulator that allows infinite number of experiences of the concept to be simulated. These simulators can be combined in various ways to create complex simulations. According to Barsalou, propositions arise as the result of binding of simulators to perceived objects in the world.

Barsalou’s proposal for perceptual symbol systems and Chandrasekaran’s proposal for multi-modal architectures have many points in common. For one, Barsalou’s system is multi-modal because the representation of a concept could consist of activations from multiple sensory-motor experiences. Further, perceptual symbols are not always converted to amodal (linguistic) symbols. Depending on the task, an agent may store and reason with perceptual rather than linguistic symbols. Perceptual symbols are also composable allowing different activations to be brought together to create (or recreate) complex experiences. This is similar to how the DRS allows diagrammatic elements to be composed to create complex shapes. However, Barsalou’s theory makes several stronger claims as to the nature of symbols. It proposes that perceptual symbols underlie even linguistic symbols, a claim that multi-modality is non-committal about. Barsalou also considers problem solving to be the search for a simulation that leads from the initial to the goal state. This is again specific compared to the more general multi-modal case where problem solving is achieved through the interleaved application of appropriate linguistic and perceptual operators on the state. Barsalou’s theory can be considered one possible multi-modal proposal albeit one whose representations and processes are biased towards explaining how cognition is achieved by the human brain.

2.6.3 Diagrammatic Reasoning Systems

From Euler and Venn Diagrams to Pierce’s existential graphs, diagrams have been used for centuries in reasoning. More recently, from Gerlenter’s Theorem Proving Machine (Gelernter) through Barwise and Etchemendy’s Hyperproof (Barwise & Etchemendy, 1996) to Lindsay’s Theorem Proving Machines (R. K. Lindsay, 1998) and Mateja Jamnik’s Diamond (Jamnik, 2001), reasoning systems have used diagrams in applications
ranging from theorem proving to general purpose representation and reasoning. Some
have also shown that diagrammatic proof techniques are just as rigorous and valid as
inferential proofs while being easier to learn and apply. Over time, diagrammatic
reasoning has been applied to tasks from planning and navigation to naïve physics
reasoning about the world. Since there are too many such systems to individually list, I’ll
consider only two – Hyperproof and GeoRep. I will provide a brief introduction to these
systems and debate whether their reasoning is multi-modal. There are two criticisms
common to both. One is that they use diagrams only for reasoning and do not have bi-
modal long term memories or learning mechanisms. The second is none of these systems
are based explicitly on a general purpose cognitive architecture. I will avoid repeating
these issues in the following discussion.

**Hyperproof** – Hyperproof is an educational tool for teaching logical reasoning using the
blocks world domain. The blocks can be cubes, tetrahedrons or dodecahedrons in small,
medium or large sizes. The blocks world is limited to an 8x8 grid. The diagrammatic part
has representations for each of these types of blocks and an 8x8 grid in which they are
placed. The symbolic part allows representation of information in standard logic.
According to Barwise and Etchemendy, “*this information will typically be compatible
with but go beyond, the incomplete information depicted in the diagram.*” Hyperproof
also provides a set of inference rules including inferences from linguistic to diagrammatic
and from diagrammatic to linguistic. The goal to be achieved, represented symbolically,
is to show that some conjecture about the given situation follows (or does not follow)
from the given information. Hyperproof uses the diagram to concretize the given
information, even creating multiple diagrams if there are multiple ways of instantiating
the given information. If a condition is true in every instance, it can be inferred as true.
Hyperproof’s advantage over standard logical inference lies in the fact that it is often
easier to apply and simpler to learn than standard logical inference.

Hyperproof clearly has representations and processes in multiple modalities, one
of which is spatial. However, due to its task domain, the information in Hyperproof’s
sentential side is always complete, which need not be true in general for multi-modal
systems, it doesn’t have to be. More importantly, Hyperproof’s perceptual (diagrammatic) representations are domain specific (cubes, tetrahedrons etc) rather than the more generic, domain-independent representations that multi-modality calls for. Hyperproof is best thought of as a bi-modal system in a limited problem domain.

**GeoRep** – GeoRep (Ferguson & Forbus, 2000) is a domain independent diagrammatic reasoning system that is based on the Metric Diagram/Place Vocabulary (MD/PV) framework. The MD/PV framework calls for two representations – one a metric diagram that preserves the spatiality of the objects (metric data) and the other a place vocabulary that is a qualitative spatial representation fitted to a particular domain and task. GeoRep elaborates on this framework by splitting the place vocabulary into two parts – a domain independent set of visual relations and a domain dependent set of task-specific visual relations. The domain independent set is called the Low-Level Relational Describer (LLRD) and the domain-dependent set is called the High-Level Relational Describer (HLRD). The input to GeoRep is a line drawing. The LLRD detects and represents propositionally, a set of early visual routines corresponding to Ullman’s universal routines. The LLRD recognizes five primitive shape types: line segments, circular arcs, circles and ellipses, splines (open and closed) and positioned text. The LLRD also combines some visual elements into polylines, polygons and glyphs. The first thing that LLRD computes over a diagram is proximity relations between the various elements. This allows GeoRep to check for other LLRD relations only between proximate elements and reduce combinatorial explosion. Once these proximity relations have been calculated, LLRD extracts orientation and frame of reference, discovers parallel lines and connection relations between those elements. The LLRD also subsumes elements into the groups mentioned earlier (polylines, polygons and glyphs). Once subsumed an element is represented as part of its composite structure instead of individually. The HLRD builds on the output of the LLRD by creating domain specific visual representations. For example, the HLRD for the task of recognizing Course of Action diagrams in the military domain would take the LLRD output and create (recognize) the various units, areas, direction of motion etc. The HLRD also contains a rule engine utilizing a logic-based
truth maintenance unit that allows it to explain why it believes particular visual elements represent particular things and a set of query routines that allow the problem solver to extract relations that may not have been extracted initially (such as those between non-proximate elements).

GeoRep in itself is not a complete reasoning system and needs to be complemented by a predicate-symbolic problem solver. Together, such a system would have representations in multiple modalities and allow reasoning to proceed by the opportunistic application of either inference or perception. GeoRep however has one drawback in that it does not allow the diagram to be modified. That is, its perceptual routines only allow the extraction of relations from the representation. This excludes reasoning that relies on making modifications to the diagram and then inspecting it to retrieve the solution. While GeoRep is not limited to a particular domain, the limited range of its perceptual operations does not allow it make full use of the perceptual modality during reasoning.
CHAPTER 3

REASONING IN BISOAR

The bimodal cognitive state described in the previous chapter has advantages for an artificial agent reasoning about the world, particularly with respect to problems that have spatial character. In this chapter, we use examples from the Blocks World domain to show how the biSoar agent has an advantage over a Soar agent when dealing with spatial problems. These problem solving examples also serve to illustrate the bimodal problem solving process. The ability to represent and manipulate spatial information also has implications for how a bimodal architecture can help with aspects of the Frame Problem (FP). In the latter part of this chapter, we describe the FP and discuss how biSoar handles FP issues.

3.1 Reasoning about Space

In the logic framework, information about the world and the objects in it is represented as sentences in a symbolic language. When a problem solving system is built in this predicate-symbolic framework, the various actions that the agent can take in the world are represented as rules that have pre-conditions that decide when the rule applies and post-conditions that explicitly capture the changes.
Consider the blocks world domain. When a block A, resting on a block B, is moved to a block C, not only does the new world have A on top of C, it also no longer has A on top of B. Further, there could be other relations that have changed, like A may have been to the left of other blocks but after the move it could be to their right. To complicate things, there may also be a lot of relations that don’t change as a result of the move, such as A’s color, size, shape etc. The relations between B and C don’t change either.

One possible way of handling this problem is to keep track of what changes are made by each action and that anything not explicitly mentioned as changing is assumed to not have changed. McDermott (McDermott, 1987) calls this the “sleeping dog” strategy. One way to implement such a strategy is using suitably parameterized add and delete lists to keep track of the consequences of actions. When the precondition of a rule was met, everything in the add list is added to memory and everything in the delete list is removed from memory. In order to control the lists from becoming too big, relations in the modeled world were divided into primitive and non-primitive relations. Non-primitive relations are those that can be inferred from primitive relations and hence, add and delete lists need contain only the changes to primitive relations.

While Add and delete lists do indeed provide a solution, they have certain drawbacks. Consider the blocks world example described above. Each add and delete list describes which primitive relations are changed by an action. As more and more primitive relations are added to the world, the number of entries in the lists also increases. At some point, the lists will grow so large that the agent spends a significant amount of time simply updating the state of the world using these lists. Another problem is that the number of inferences required to derive a non-primitive relation from the primitive relations may turn out to be expensive and repeated application of such inferences could slow down the system.

3.1.1 External Representations

Consider the same blocks world scenario as before except that the agent now has a piece of paper and a pencil and the ability to draw and erase shapes on the paper. Instead of
representing blocks A, B and C using predicates, the agent instead draws them as blocks on the paper. If the agent needs to know the relationship between any of the blocks, it simply looks at the diagram and extracts the required information from it. If the agent has to move block A from B to C, it simply erases block A from its previous position above B in the diagram and redraws it on top of C. One can see how there is no need for add and delete lists to keep track of primitive relations. Now consider the example of adding a new relation right-of to the vocabulary of the world. This would involve adding perception and action routines that tell the agent how to check for the right-of relation and how to move something to the right-of another respectively. And that’s all. There is no need to modify any other relation.

What makes this form of representation so powerful is a combination of factors – One, the nature of the problem allows the use of a spatial representation. There are many problems that can’t be transformed into such a representation and cannot be solved using this technique. Two, the structure of the physical world ensures that the causality of space is applied to the diagrammatic representation. Three, the perceptual abilities of the agent are capable of carrying out the tasks that are required for perceiving, creating and modifying diagrams. This ability of diagrammatic representations (and spatial representations in general) to make explicit certain implicit consequences of an operation, has been referred to variously as free rides and emergent properties (B. Chandrasekaran et al., 2004).

While the use of external analog representations is non-controversial, there has been much debate about the presence and availability of internal analog representations for reasoning. Without getting into the debate, our internalization of this representation can be justified simply as an AI solution to an AI problem.

### 3.2 Blocks World in BiSoar

#### 3.2.1 Example 1

Let us start with an extremely simple example, Fig 3.1(a), to illustrate the basic ideas and issues. The situation has only two blocks – A and B and a Table, one relation on-top-of and a move-on-top-of operator. The goal is to create a domain state that satisfies the
description ON(A,Table). We will run through the representations in Soar, describing its problem space and working memory at each point in problem solving, and repeat the sequence for biSoar.

Fig 3.1(b) shows the starting state of working memory in Soar. It contains a description of the world state, and the current goal. During the proposal phase, the production for proposing a move operator fires and a move operator is proposed to move A onto the Table. Fig 3.1(c) shows the state of Soar after the operator proposal phase. During the application phase, shown in Fig 3.1(d), the rule for applying the move operator fires and removes On(A,B) from the state and adds On(A,Table) to it.

Fig 3.2(a) shows the starting state for biSoar. There are two blocks – A and B and a table T. The problem state now has a diagram, represented in DRS, as a component. The goal to be achieved can be represented both symbolically and diagrammatically, but because of the ambiguity inherent in diagrammatic representations about what is intended, we only use symbolic goal descriptions. Unlike in standard Soar, in biSoar there is no requirement for the symbolic part of the state to contain predicates describing the initial state world state, if the diagrammatic component depicts the situation. During the proposal phase, the rule that proposes the Move operator fires (this state is not shown in any of the figures). During the application phase, instead of updating the symbolic part, the rule calls the action routine to update the diagram to reflect Move(A,Table). Checking for preconditions can be done directly by the relevant perceptual routines. Fig 3.2(b) shows the final state after the move operator has been applied. Unlike standard Soar, biSoar does not need add or delete lists to keep track of the state of the world. The diagrammatic part does it instead.

\[1\] For our purposes, a content description of Soar’s WM is all that is required. This is what the figures represent and should not be mistaken for an exact replica of Soar’s WM.
Figure 3.1: (a) Simple BW example (b) Initial contents of Soar’s WM (c) Soar’s WM after operator proposal (d) Soar’s WM after Move applied
3.2.2 Example 2
Let us add the following new relations to the world: under, above, below, imm-right-of, imm-left-of, right-of, left-of and inside-of. These relations are interpreted in their natural meanings, so we forego formal descriptions of them. The goal state to be achieved is described in terms of above and right-of relations. For our list of relations, on-top-of, under, imm-right-of, imm-left-of and inside-of are the primitive relations while above, below, right-of and left-of are the non-primitive relations. In Soar, the primitive relations, from which all other relations may be derived, are updated after each change in the world. If a non-primitive relation is needed, the solver performs inference to find the answer.
Fig 3.3 shows the initial state for our blocks world problem. The final state is laid out as a sequence of goals to be achieved by the problem solver, while the initial state is simply a DRS representation of Fig 3.3. The goals to be achieved are: B inside-of B1, E above A, F above A, H above A, D above B, G above A. The problem solving sequences for the standard Soar problem solver and bi-modal Soar are shown in Fig 3.4. The solvers try to achieve each goal in the sequence in which it is presented. We examine one slice of this sequence. Consider the final sub-goal of the problem “G above A”. To achieve this sub-goal, standard Soar first checks if Block A is clear. Since it’s not, the solver sets up a sub-goal to find the topmost block above A. In order to find the topmost block, the solver performs inference by moving up the stack starting with block A. It finds that E is on-top-of A, that F is on-top-of E, H is on-top-of F and that there is nothing on-top-of H. Instead of just 3 blocks above A, if the stack had 20, the solver would have had to go through 20 such steps to find the topmost block. Consider the same sub-goal being solved in bi-modal Soar. The sequence does not vary from any of the other sub-goals. The solver calls a perceptual routine to check if A is clear. Since A is not, it calls the perceptual routine topmost to find the topmost block above A. The routine returns H and the solver calls the move routine to move G onto H. This sequence of problem solving steps is independent of the number of blocks in the stack. If there were 20, the solver would still call the topmost routine just once to find the topmost block.

Adding these new relations also means that we have to add the corresponding move operators for each of these relations as well as modify the existing move-on-top-of operator. For example, consider adding the imm-right-of relation. The corresponding

![Figure 3.3: Initial state for the blocks world example](image-url)
move-imm-right-of operator will update the state of the world by adding and deleting the appropriate imm-right-off() predicates. It will also have to maybe add and delete some ON predicates depending on whether the block being moved was on or is being moved on to a block. But this is not enough. We also need to modify the existing move-on-top-of operator, because now, moving a block on top of another block could change its imm-right-of relations with other blocks. Similarly, now adding imm-left-of means that we need to modify both move-on-top-of and move-imm-right-of operators.

<table>
<thead>
<tr>
<th>Standard Soar</th>
<th>Multi-modal Soar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check if Box B1 clear</td>
<td>Check if box B1 clear</td>
</tr>
<tr>
<td>Move E to inside of B1</td>
<td>Move E to inside of B1</td>
</tr>
<tr>
<td>Check if A is clear</td>
<td>Find topmost block above A</td>
</tr>
<tr>
<td>Move F onto E</td>
<td>Move F onto A</td>
</tr>
<tr>
<td>Check if A is clear</td>
<td>Find topmost block above B</td>
</tr>
<tr>
<td>No. Find block on-top-of A</td>
<td>Find topmost block above A</td>
</tr>
<tr>
<td>Check if E is clear</td>
<td>Move H onto A</td>
</tr>
<tr>
<td>No. Find block on-top-of E</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>Check if F is clear</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Move H onto left-half of F</td>
<td>Move D onto F</td>
</tr>
<tr>
<td>Check if B is clear</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>No. Find block on-top-of B</td>
<td>Move D onto F</td>
</tr>
<tr>
<td>Check if E is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>No. Find block on-top-of E</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Check if F is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>Move D onto right-half of F</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Check if A is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>No. Find block on-top-of A</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Check if E is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>No. Find block on-top-of E</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Check if F is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>No. Find block on-top-of F</td>
<td>Find topmost block above E</td>
</tr>
<tr>
<td>Check if H is clear</td>
<td>Move H onto F</td>
</tr>
<tr>
<td>Move C onto H</td>
<td>Move C onto H</td>
</tr>
</tbody>
</table>

Figure 3.4: The problem solving sequences for standard and multi-modal Soar for the problem in Figure 3.3.
In biSoar, instead of modifying the symbolic content, we add perceptual and action routines corresponding to the *imm-right-of* and *imm-left-of* to the diagrammatic component. The *move-on-top-of* operator however was left untouched. According to Janlert (Janlert, 1996) “A sign that the frame problem is under proper control is that the representation can be incrementally extended: A conservative addition to the furniture of the world would involve only a conservative addition to the representation.” In our case, the world is the blocks world and an addition to the world can be in the form of objects and/or relations. The examples show that biSoar handles both additions well without exponential additions to the agent or modifications to its knowledge of existing objects and relations.

### 3.2.3 Evaluation

Diagrammatic reasoning is not always the best approach to solving problems; even those that may be spatial. What we might call the *relative cost rule* applies: It is practical only when the cost of obtaining the needed information by applying perception is less than the alternative, viz., performing inference on a symbolic model of the situation. Let us first consider the situation where the agent can apply perception to the relevant external world or external representation. The time cost of applying perception is the sum of the costs of visual search to locate the objects that are the arguments of the perception in question, and of the perception itself. If this time cost is less than the time required to apply the relevant sequence of inferences (in the case of Soar, the number of decision cycles required to carry out the inferences), perception is cheaper. For example, humans, faced with climbing on ladders to search through hundreds of blocks, might well decide to try to infer than get the information from the external world. The human architecture can carry out some perceptions at about the same time cost as one decision cycle, so we can predict when people would find it more convenient to access external representations visually than reason about the information needed. Conversely, the perception may not be available to the agent at all, e.g., checking to see if block A is .05 mm longer than block B. There is also the issue of short term memory -- if reasoning is likely to stress STM,
whereas perception can pick up the information without stress on memory, the agent would prefer perception.

If the choice is between internal perception on an internal image and inference on symbol structures, the calculus is more complicated and uncertain. The trade-off would depend on the relative costs of multi-modally distributed STM and the degree to which internal images and operations on them mimic the structure of physical space. For artificial agents, the same relative costs rule applies, but the actual parameters would differ not only from the case for human agents, but from one specific artificial agent to another. For one artificial agent a particular perception may have a total computational cost less than inference, while for another, the opposite may be true.

Since traditional approaches to representing the state of the world in an agent involve a trade-off between space and time, we use these as our criteria for evaluating Soar and biSoar. If an agent were to represent every relevant proposition about the world, the time to infer a relation would be a constant, but the space required would rise exponentially as the number of relations grows. On the other hand, if the agent were to represent only certain relations, called primitive relations, from which all other relations can be derived, the agent saves space but has to perform time-consuming inference. Due to space considerations, the latter approach usually wins out and is used in the Soar agent in the example above. However, as the examples also show, it is possible for a diagrammatic representation to be fairly compact while still being able to extract information about state of world without having to perform inference. We now provide a quantitative space-time comparison between Soar and biSoar.²

²The diagram in the DRS is often incomplete (with respect to an external diagram or the world itself) in the sense that its contents are determined by various factors such as the current goal and attention, to mention a few. Thus, the contents of the DRS are often limited to the objects relevant to the perception being performed. However, for the current purpose of space analysis, this distinction is not important since it can be argued that the same is applicable to the predicate-symbolic side.
Table 3.1: Space requirements for Soar and biSoar

<table>
<thead>
<tr>
<th>Agent architecture</th>
<th>Space requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soar</td>
<td>$n+rn$</td>
</tr>
<tr>
<td>biSoar</td>
<td>$n+2kn$</td>
</tr>
</tbody>
</table>

Table 3.1 summarizes the space requirements for Soar and biSoar for representing the state of the world in the blocks world domain. For Soar, the space requirement of $(n+rn)$ comes from two factors - Representing $n$ objects and $r$ primitive relations for each object. It is assumed that each object is a primitive relation with at most one other object. If not, the space required is even greater (by a factor of $n$). For biSoar, there are $n$ representations on the symbolic side, one for each object. On the diagrammatic side, currently each object representation requires $k$ $(x,y)$ coordinates for a total of $2kn$ coordinates for $n$ objects.

The space requirements can vary, especially for the diagrammatic representation where an object can be represented in a variety of ways – by a set of coordinates such as in the current implementation, an array representation, an equation etc. However, irrespective of the implementation method, the diagrammatic representation is independent of the number of primitive relations recognized in the domain. For current purposes, we can see that the diagrammatic state representation is not much worse than the symbolic one. In fact, if the number of primitive relations were increased, the space requirements for Soar would propositionally increase while biSoar would be unaffected. On the flip side, space requirements for biSoar would increase if arbitrary shapes, involving many coordinate points per shape, were to be used.

Table 3.2 shows the time requirements for Soar and biSoar. Soar and biSoar are compared along two dimensions – the number of Soar cycles spent on performing an inference (for example, to find the topmost block above a block) and the number of Soar working memory elements to be updated after an action is executed (for example, after a
block A is moved from block B to block C, the state has to be updated to show that A is no longer on B, but on C). In the following, we assume that the time cost of perceptions of interest is about the same order as the cost of one decision cycle, as happens to be the case for the human architecture.

<table>
<thead>
<tr>
<th>Best Case (1 relation, 2 blocks)</th>
<th>Soar Cycles</th>
<th>Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soar</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>biSoar</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Worst Case (r primitive relations, n blocks)</th>
<th>Soar Cycles</th>
<th>Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soar</td>
<td>$n-1$</td>
<td>$4r$</td>
</tr>
<tr>
<td>biSoar</td>
<td>1</td>
<td>None</td>
</tr>
</tbody>
</table>

Table 3.2: Time requirements for Soar & biSoar

In the best case, the diagram has only 1 relation and 2 blocks. Soar (and biSoar) take 1 soar cycle each to infer (basically retrieve) the relation between the blocks. Additionally, Soar doesn’t have any updates to make if the blocks were placed in such a way that no relations hold before or after the move. biSoar never has to make updates because changes are made to the diagram itself. In the worst case, Soar takes $n-1$ cycles to infer a non-primitive relation. For example, inferring the topmost block on a stack of n blocks starting at the lowest block and chaining the on-top-of relations would take $n-1$ cycles. Soar also needs to make $4r$ updates where $r$ is the number of primitive relations. The basic action in the blocks world is moving a block, which involves deleting its current relation attributes and adding the new ones. Since there are $r$ relations, that involves $2r$ updates. Also, the block would have been in at most $r$ relations with blocks around it.
(assuming that multiple blocks can’t hold the same relation to a block). Thus, there would be \( r \) blocks with a relation each that needs to be deleted. Similarly, there would be \( r \) new blocks with a new relation that needs to be added.

One caveat about the above analysis is that it assumes a fixed cost of a single inference cycle for perception. The true cost of perception, however, depends upon the perception being performed and the state of the environment on which the perception is performed. For example, finding one object from among 2 or 3 objects is faster than finding that object from among 20. This does not mean that there is an advantage only when the perceptions are faster than the equivalent inference steps. Perceptual machinery, as in the case of humans, can be operated in parallel to the inference machinery. This parallelism can be exploited to achieve speedup during problem solving.

### 3.3 The Frame Problem

As formulated by McCarthy and Hayes (now referred to as the logical or technical aspect of the FP), the FP is the problem of axiomatizing the *Common Sense Law of Inertia* - the understanding that an action is assumed to not have changed a property of a situation unless there is evidence to the contrary (Shanahan, 2004). A different problem involves how to reason about an action’s consequences without having to go over the entirety of its (the agent’s) knowledge. McDermott calls this the computational aspect of the FP (McDermott, 1987). Later discussions have identified other distinct, but related problems, including the Philosopher’s aspect of the FP and the Ramification Problem, lurking where just one problem was first seen. There is broad consensus today that the frame problem in its logical guise has been solved (Shanahan, 1997) (Reiter, 2001). However, the computational aspect still remains problematic. For the remainder of this chapter, when we refer to the FP, we mean its computational aspect.

As mentioned earlier, in the logic framework, information about the world and the objects in it is represented as sentences in a symbolic language, and the various actions that the agent can take in the world are represented as rules that have pre-conditions that decide when the rule applies and post-conditions that explicitly capture the changes. The FP is faced by an agent reasoning in a dynamic world. The simple blocks world example
used in Section 3.1 illustrates the FP. When an action is taken in the world (moving A to C), it has both direct (A is now on C) and indirect consequences (A is to right of D), and the Frame Problem is the problem of having to keep track of the indirect consequences without having to reason about each and every relation in the world. Hayes (Hayes, 1987) provides a more comprehensive introduction to the frame problem. The “sleeping dog” strategy, via the use of “add” and “delete” lists, is considered the most effective way of handling the FP. However, while add and delete lists do indeed provide a solution to the FP, they have the drawbacks mentioned earlier – the large size of these lists and consequent performance losses, and the inferences required to derive non-primitive relations. These drawbacks are well documented in the FP literature.

While various heuristics and strategies might mitigate the problem, there is a consensus that the FP and its variants are unavoidable in the sentential knowledge representation framework. This has resulted in suggestions that perhaps alternative representative frameworks might exist that avoid the problem. One such proposal by Janlert (Janlert) is that analog representations such as pictorial representations might be the answer. In such systems, a rule only encodes the basic action. The other changes are implicitly captured by the representation (as in the external world). Unfortunately, analog representations suffer from problems too, one of which is their tendency to over-specify. Due to this, the possibility exists that conclusions drawn within the framework could be wrong. To avoid this, agents using such representations usually perform additional reasoning that verifies any conclusions. As Pylyshyn (Zenon W. Pylyshyn, 1996) has mentioned, when this additional reasoning is combined with the problem of converting problems into equivalent ones that have a spatial character, we end up trading one problem for another equally difficult problem.

While reformulating every problem into a spatial one has prohibitive costs and is not always possible, we believe there is a small subset of problems, namely those that are already spatial or have a natural spatial analog that have no need for conversion. Analog representations can indeed be of help in these cases. Secondly, analog representations have another advantage. The sleeping dog strategy, the preferred technique for handling the problem in the logic framework, has the drawback that any addition to the vocabulary
of operations in the world results in changes to the existing knowledge in the system. A system with access to a diagrammatic representation does not have the same problem.

The advantages are clear in the case of an external representation, such as a diagram, for keeping track of the world with there being no add and delete lists to update after a change in the world. Similarly, the addition of new relations merely involves defining new perceptions. There is no need to modify any other relation.

biSoar’s diagrammatic WM allows an agent to represent a diagram internally. Thus, the geometrical benefits of an external representation are available to the biSoar agent as part of its cognitive state during problem solving. This idea of diagrammatic representations as a solution to the frame problem has been proposed before, most notably by Lindsay (R. Lindsay, 1995). But, while Lindsay does lay out his vision of such a diagrammatic system, he merely mentions that “One may view perception as offering a solution to the frame problem by allowing “the world” to make appropriate inferences which are then “read” by the brain/mind.” Our work takes a closer look at the frame problem space and identifies exactly where diagrammatic representations can make a contribution. Also, we propose diagrammatic representations as a solution to the drawback of the sleeping dog strategy and lastly, we show how a bimodal cognitive architecture has the capability to solve such problems.
The architecture that we have described in the previous chapter, biSoar, is a model of high-level cognition. The experimental results that we seek to explain are, however, about a complete agent interacting with external representations, such as maps. In our proposal, part of the explanation of the results arises from the perceptual system and its interface with central cognition. In this section, we provide a high-level overview of a complete biSoar agent that can interact with the world, and identify the phenomena that are relevant to explaining our results. I then describe an interface module and external world representation that captures the relevant phenomena and allows biSoar agents to behave as if they are interacting with the world without the need for a perceptual system.

4.1 Interacting with the External World

Most cognitive architectures such as Soar and ACT-R model central cognition. An agent built in these architectures cannot interact with the physical world without a collection of perceptual and motor modules, the former to process sensory data and the latter to act on the world. Physical agents built in the framework of any of these architectures make use of available perceptual and motor modules, but the theories underlying their processing
are not part of such cognitive architectures. However, these architectures may have specific proposals about the interfaces with these modules, for example, ACT-R’s visual buffers. In such an agent, the contents of WM representing the external world are the result of a combination of perception and attention processes. Attention, as the word implies, is selective: it places in working memory selected aspects of the results of the perceptual module. What is selected -- attended to -- depends on the agent’s goals. Figure 4.1 gives a high-level overview of the components in a complete agent.

![Figure 4.1: A high-level overview of the components of a complete agent](image)

biSoar includes, as part of its central cognition, representations that are modality-specific, but, like Soar, lacks external sensory processing. A biSoar agent interacting with an external representation would require, similar to Soar, the appropriate perceptual modules that process visual information, and motor capabilities to change the external representation. In addition to producing symbolic information, as in the case of Soar, the perceptual system for biSoar would produce DRS representations corresponding to selected parts of the external representation.
4.1.1 Role of Attention

The net result of the perception and attention processes is that only some of the information that is available for perception ends up in central cognition. In the case of Soar, the result of the perception-attention combination is that only a subset of the symbolic information available about the external world ends up in WM. With biSoar, which has both symbolic and diagrammatic representations, the result of perception and attention is that only some of the symbolic and diagrammatic information available enters WM. Thus, there is a loss between the information available on the retina and that which ends up in WM. In the diagrammatic case, this loss of information can be manifested in two ways – it could result in a subset of the diagrammatic objects available in the representation ending up in diagrammatic WM, and/or, it could result in a loss of some of the details of the spatiality of an attended to object. Such loss of spatiality has been noted, initially by Gibson (Gibson, 1928) who commented that “curved lines are much more apt to be reproduced as straight lines than the reverse”, and later by others including Tversky (Tversky, 2005b).

The extent of the loss, as in the symbolic case, depends to a great degree on the goals of the agent. As an example, consider a route-finding task given a map, where an agent might be concerned only with the general curvature of routes in the map. When the agent, with this goal, observes a curve standing for a route in the map, the output of the
perception-attention process is a simplified representation of that curve. This simplified representation retains the general spatiality of the original but loses some of the detail. Fig 4.3 illustrates this effect.

**Figure 4.3: A curve and the effect of detail loss on the curve.**

In cases where the details of an object are, in fact, important to the task, the agent can limit the amount of detail loss by attending to smaller regions in the external representation, but, the entire process may require multiple episodes of attention. The smaller the region that is attended to, the lower the loss of detail. As an example of this process, let the agent have the task of remembering the region shown in Fig 4.4(a). When the agent pays attention to the entire object, it would have Fig 4.4(b) in WM. This memory of the object is not enough to recall the object in detail. To limit the loss of detail, the agent attends to pieces of the original object, one at a time and in some sequence. When the agent attends to a piece, it is present in WM and put away into LTM.

3 A recall task in Soar is not as straightforward as it seems, due to the recognize-act nature of Soar’s rule based LTM. For example, if the task is to recall B given A, then the initial rule learned looks like this: “If the task is a recall task and A & B are given, then recall B”, which is useless when the agent is given A (since it requires B to be given too). This problem, called data chunking, can be handled through a generate-test approach in recall tasks. Newell (ref) gives a more detailed explanation of the problem and its solution. For the purposes of our example, we set aside the data chunking problem.
This process is repeated till the agent has learned all of the pieces. Fig 4.4(c) shows the various pieces of the object in Fig 4.4(a) that the agent learns. The agent determines what constitutes a piece by attending to various salient features of the curve. The number of pieces and the sequence in which it is remembered is dependent on the complexity of the object.

4.1.2 Proprioceptive Information

As illustrated in Fig 4.1, at any problem solving instant, an agent usually only focuses on a sub-region of the external representation, and it is from this sub-region that the representation in WM is constructed via the perception-attention processes. When the agent attends to a different sub-region, WM changes accordingly. At the end of problem solving, the agent has constructed, in WM, a series of diagrams (and symbolic information) from various sub-regions of the external representation. Assuming that these diagrams were put away in LTM during the course of problem solving, the agent can recreate the external representation if faced with a task that requires such recreation. However, it has available only the individual diagrams and has to recreate the representation by appropriately aligning these diagrams. This requires that the agent

![Figure 4.4: An example where the agent needs to recall the exact spatiality of curves](image)
have some information about where each diagram belongs, i.e., where in the external representation the agent was focused when the diagram was constructed. In certain cases, this information can be symbolic, but, in many cases, symbolic information is too imprecise to account for human ability. Consider an agent recreating Fig 4.4(a) from the pieces in Fig 4.4(c). It is not enough to simply know which pieces are connected to which, but it is also necessary to know that a piece is in particular spatial relation to another piece.

This requires that there be some form of non-symbolic location information that is available to agent, and can be learned, recalled and used during recreation. One possible source of such information is proprioceptive memory. In humans, when attention is switched to a different part of the external representation, our eyes, head and body move appropriately to allow us to focus on the new section. This proprioceptive memory is available to cognition and can form the basis for the region’s location (Wiener & Taube, 2005). The experience of an object’s location independent of the perception of the object is a feature of many theories of the human visual system. Marr (Marr, 1982) refers to this distinction as one between the location of an object and its type. Ullman refers to the memory of location information as Marking (Ullman, 1996), and underscores its role in abilities as fundamental as visual counting. Proprioceptive memory also forms the source of information for Pylyshyn’s Location Indexes (Z. W. Pylyshyn, 1989). Lastly, there is support from neurophysiology, where studies have shown the existence of independent pathways for object location (where) and object identification (what) (Mishkin et al., 1982).

A sufficiently precise proprioceptive memory should allow for near perfect recall of the external representation. However, experimental studies in humans have shown that people are unable to place an object in its exact original location. This error in identifying the original location decreases if additional reference information is provided (for example, if many objects are shown and one is taken away, people are much better at putting the object back in its original location.)

This phenomenon of location inexactness follows a pattern of information loss during learning, whether it is the loss of detail in the perceptual-attention system or the loss of information during chunking.
has been well studied in the literature. Nelson and Chaiklin (Nelson & Chaiklin, 1980) and Newcombe et. al. (Newcombe et al., 1999) among others have reported on biases in the recall of point locations. Recalled locations are biased by a variety of factors including the presence or absence of boundaries or barriers and reference points.

Our overall comprehension of the space around us is, thus, built up from individual WM diagrams of the attended sections of the external representation, associated location information arising from proprioceptive memory, and symbolic information about the objects and the relationships between them. During recall, depending on the situation, the agent uses the available information to recreate the original representation. For example, when recreating the object from Fig 4.4(c), the agent has available – the three pieces corresponding to the three sections (let’s call them c1, c2 and c3) and the symbolic knowledge that the pieces are touching each other to form a curve. The agent is free to place the first piece anywhere (let’s assume that the agent chooses to start with piece c1). To place c2, it utilizes the constraint provided by the knowledge that the starting point of c2 is the end point of c1. To place c3, it again uses the knowledge that the end point of c2 is the start point of c3. In the (unlikely) case that the agent recalls c1 followed by c3, it has three constraints for where and how to place c2 – the start and end points as given by the end point of c1 and the start point of C3 respectively, and the fact that the pieces form a curve. The agent has to appropriately scale c2 to ensure that the continuity of the curve is maintained. This process of using a combination of diagrammatic, location and symbolic information is not limited to recreating a previous external representation. In fact, it can be used to imagine new and original representations that combine individual diagrams in different ways.

The inexactness of diagram location due to the nature of proprioceptive information and information loss via the perception-attention process result in diagrammatic objects being relocated and simplified during recall. Fig 4.5 shows the consequences of these effects on the three types of diagrammatic objects.

While the losses may be due to limitations of the human architecture, it is possible that such loss has a more functional advantage for intelligent behavior. One possibility is that the loss of detail allows the agent to retrieve items faster from memory with the additional overhead of interacting with the environment to recover from the loss of detail, while using the environment itself to fill in the details of the exact location, once attention is directed to a place that is sufficiently close.
4.2 An Interface between biSoar and the External World

Our theory is about how biSoar forms spatial memories. The experimental results that we seek to explain are about a complete agent interacting with external representations, such as maps. However, in order to avoid implementing a perceptual module that processes external representations, we adopt an implementation strategy that is functionally equivalent to an agent interacting with a physical representation. We encode the external representation as a data structure that is itself implemented in DRS, henceforth denoted by DRSext. DRSext is a diagrammatic representation of the external world that is schematized with respect to the kind of tasks that biSoar expects to perform. biSoar agent’s are provided with an interface to DRSext that allows the agent to specify which object or objects it is interested in. biSoar’s attending to DRSext produces in WM the same DRS that the same agent would produce if it were given a perceptual module and attended to a sub-region of the external representation. Fig 4.6 illustrates the current implementation.

Figure 4.5: The effect of location inexactness and detail loss on (a) point (b) curve and (c) region objects. Original objects are black and recalled objects are red.
The interface also implements the loss of detail phenomena detailed earlier using the Discrete Curve Evolution algorithm (Latecki & Lakämper, 1999). The consequence of executing the DCE algorithm on a diagrammatic object is the elimination of minor perturbations in the object’s spatial extent. This loss of information about the curve’s spatiality approximates the loss of information suffered during human perception. Whenever an object is accessed from DRSext, the interface ensures that the DCE algorithm is run on the object before it is passed on to biSoar. The algorithm can be parameterized to vary the amount of detail loss. The DCE algorithm is only one of many algorithms that can be used to mimic the loss of detail. The algorithm’s simplicity and ease-of-parameterization made it our choice. It is possible that there are other algorithms, maybe even biologically inspired in their operation, that are better suited for the task. For now, the selection is based on convenience rather than any commitment to its accuracy or biological validity.

Finally, the interface also passes on location information for each region that the agent focuses on. This location information is represented as a point that stands for the center of the region where the agent’s focus lies. During recall, one of the constraints available to the agent is that the recalled objects lie within this region. The dimensions of this region where the diagram is likely to be located can also be set via a parameter.
CHAPTER 5

MODELING LARGE-SCALE SPACE

For the rest of the thesis, we concentrate on the application of the biSoar architecture as described in Chapters 2 and 4 to the task of modeling human performance in spatial reasoning. We restrict our efforts to a particular subset of the overall spatial reasoning problem - reasoning about large-scale space. In this chapter, we elaborate on the domain, identify certain general suitability criteria that any modeling system should satisfy before it is ready to model in this domain and show, using examples, how biSoar satisfies all of the criteria. We then examine related work in this domain. Before starting out, we explain what we hope to accomplish by applying biSoar to the modeling problem.

5.1 Goals of Modeling Spatial Reasoning in biSoar

The arguments in favor of cognitive modeling as a methodology for the study of cognitive phenomena have already been made by others (Cassimatis & Bello, 2007) (Ritter et al., 2001) and, Newell (A. Newell, 1990) has extensively remarked about the advantages of the cognitive architecture approach in the modeling process. The merits
and demerits of diagrammatic reasoning have also been talked about in the previous chapter. We elaborate on what we hope can be achieved currently by using biSoar as a modeling tool.

**Classifying Explanations** - Cognitive modeling allows researchers to create different models of a phenomenon by varying representations and parameters. To the extent that a model is accurate, researcher’s knowledge about the internal structure and representations of the model allow them to draw conclusions, at some level of abstraction, about human performance in the same task. When models are implemented in a cognitive architecture as possible explanations for a phenomenon, the behavior of interest can be said to arise from one, or a combination, of two influences - Architecture and Content, where Content is simply the agent’s knowledge. Even though Soar (and biSoar) do not distinguish between different kinds of knowledge – strategic vs factual – it is useful to think of the information in LTM in this way. Content can, thus, be further sub-divided into strategic knowledge and factual knowledge.

An architectural explanation appeals to the properties of the agent’s architecture to explain the phenomenon of interest. In the case of such an explanation, the phenomenon emerges as a consequence of architectural properties rather than particular strategies or knowledge employed by the agent. A phenomenon can also emerge as a result of a particular strategy employed by the agent to solve the given task. The explanation in this situation is different from an architectural explanation because the phenomenon is unique to the current strategy. An agent’s behavior can also be seen as arising from its knowledge (or lack thereof) of the task domain and the world. During problem solving, an agent may learn to solve the problem one way due to the knowledge it has at the time. Given more knowledge, the agent might have learned to solve the problem in a different way, resulting in different observable phenomena.

Once an explanation is classified into one or combinations of the three types, it is possible to predict certain consequences based on the type of explanation. For instance, if an architectural explanation is proposed for a phenomenon, then that phenomenon should be present across a variety of agents. This is because an architectural explanation appeals
to the particularities of the architecture which is common across all agents (based on that architecture.) Take, for example, Soar’s learning mechanism, chunking. Chunking is an architectural learning mechanism, with one of its claims being that agents learn automatically without deliberative effort. This claim is borne out by the fact that at least some of human learning is involuntary and a result of experience. If an architectural explanation is proposed, but general coverage is not found, then it is likely that it is not the correct explanation. Alternatively, if the explanation appealed to strategy or knowledge, it should be possible to manipulate the phenomenon by changing either the strategy used or the knowledge available to an agent.

In general, a phenomenon can have more than one explanation, and it is difficult for a theorist to decide if the reason for the phenomenon is architectural-, strategic- or knowledge-related without further experimentation. Also, due to the number of free variables and tunable parameters in cognitive architectures, the ability (or inability) to build a model in the architecture cannot be taken as the final word on whether the explanation offered by the model is correct (or incorrect). Under certain circumstances, however, the inability to build a model in the general cognitive architecture framework can be taken as a sign that the model is flawed.

**Identifying Control Variables** - Building models gives us another advantage. It provides us with a list of possible explanations for the phenomenon which can then be used to develop a series of controlled experiments that can decide between the various explanations. As we will later show, models in biSoar have a straightforward mapping to issues to control for, and building these models provides a natural way of discovering these issues. Of course, the experimenter is free to simply think of various explanations without modeling in biSoar, but the advantage of using biSoar is that it provides additional constraints on the possibilities and restricts the experimenter to those explanations that are cognitively possible. In addition, a biSoar model can work out the consequence of interactions better than the theorist can do, much like a physicist’s explanatory hypothesis, written down as a differential equation can be simulated computationally and consequences identified. The experimenter/modeler is still required
to form hypotheses for model building as there is no systematic way of generating these models. Certain heuristics such as “look for at least one explanation from each possibility in the explanation space” can suggest lines along which the model builder/experimenter may approach the problem.

**Investigate Contributions of Individual Strategies and Mechanisms** - A final advantage of the architectural modeling approach is that it makes it possible test the contributions of individual strategies or mechanisms to the phenomenon of interest. In the case of human subject experiments, it is impossible to start with a clean slate. Each subject brings to the table various personal perspectives and idiosyncrasies that color his or her performance. Clever experiment design and larger test groups can eliminate problems in some cases, but there is enough shared knowledge and stimuli that make it hard to tease out the roles of individual strategies and mechanisms in producing the phenomenon. On the other hand, it is easy to exactly specify what kind of information a biSoar agent has and control what it has access to, as well as examine the agent to see the source of the phenomenon. In one of the examples in chapter 5, we show how biSoar can be used to test one hypothesis for a phenomenon involving the influence of spatial goals. In an earlier study, Taylor, Naylor and Chechile (Taylor et al., 1999) found that the task used during learning affected subjects’ performance in the subsequent recall task. The conclusion was that the subjects’ representation formed in memory was different depending on the learning task. We built simple biSoar models that show how such an explanation can be tested within the biSoar framework.

We illustrate these points about biSoar by using it to model cognition involving large-scale space.

### 5.2 Modeling Large-Scale Space

As mentioned in the introduction, the term cognitive map was first coined by Tolman to refer to the representation of large-scale space in rats that allowed them to remember and navigate back to food sources in a maze. In 1960, Lynch produced his seminal study of the environment and its features that are important in building a cognitive map (Lynch,
Lynch identified *landmarks* – salient cues in the environment such as distinctive buildings, *routes* such as roads, rails and even bike pathways that connect various landmarks, junctions or intersections of routes called *nodes, districts* which are implicit or explicit regions of the city, and *edges* that prevented travel, demarcated the different regions and bounded the city itself. The longest standing model of large-scale space acquisition and representation is the Landmark, Route, Survey (or LRS) model (Siegel & White, 1975). LRS theory states that an agent first identifies landmarks in an environment, adds route knowledge between landmarks as he/she traverses the environment and finally adds survey (or configurational) knowledge as the agent becomes familiar with the environment. Once survey knowledge has been added, the agent has the capability to propose novel, previously un-traversed paths between landmarks. In 1978, Stevens and Coupe proposed a hierarchical model of spatial memory to account for distortions in judgments of relative geographical locations (Stevens & Coupe, 1978). For example, when subjects were asked if San Diego, CA was to the west of Reno, NV, a number of them said yes even though this answer is incorrect. Stevens and Coupe hypothesized that subjects stored spatial information about cities and states as hierarchies. Errors in judgment occurred because relation information is not stored at every level, and subjects wrongly inferred the relationship between cities using the relation of the corresponding super-ordinate political units. Later theories have modified these models in various ways. For example, in 1998 Gilner and Mallot proposed the view-graph theory, in which views and egocentric vectors replaced places (view-independent) and allocentric vectors in the cognitive map (Gillner & Mallot, 1998).

### 5.2.1 Sources of Knowledge for the Cognitive Map

There are three potential sources of information for an agent’s cognitive map (Newcombe, 1985). The primary source is, of course, the agent’s experience of navigating in the environment. The environment can be a real one, such as building or city (Lynch, 1960), or it can be virtual, where the agent is allowed to explore the space on a computer via some input device (T. P McNamara, 1986). There are two secondary sources of knowledge – maps and linguistic information. Both (though maps more than
linguistic information) are usually the only source of information that covers large areas such as the layout of states or countries in the world. Maps are ubiquitous in everyday life, especially with the availability of online mapping tools and GPS navigation systems. Linguistic or verbal information is usually the most ambiguous and incomplete source of large-scale space information (though it need not be). Still, verbal descriptions of routes and relationships between objects in the world constitute an important source of information in the cognitive map creation process (Tversky, 2005a).

While most studies involve learning along one of the above sources, in real life, our cognitive maps are built up from a combination of these sources. However, for the purposes of this thesis, only maps and linguistic/verbal information are used as sources of knowledge. While this leaves out the primary source of information, studies such as those by Tversky (Tversky, 1992) have shown that the phenomena modeled in this thesis are exhibited by agents irrespective of whether the cognitive map was formed by navigating in the world or viewing a map of it.

5.2.2 Testing the Cognitive Map
Taylor (Taylor, 2005) identifies three different types of tests that experimenters commonly perform in cognitive map experiments. The simplest and most straightforward method is to ask the subject to sketch a map of the test area. Lynch uses this method to great success in his study of subjects’ cognitive maps of Boston. The advantage of this method is that it provides a direct estimation of the cognitive map but with the drawback that the results are often influenced by the drawing abilities of the participants. Another set of tests involves estimation tasks such as pair-wise judgment tasks. In such tests, once the agent has learned a representation of an area, he or she is asked questions such as “Estimate the distance between locations A and B on the map” or “Estimate whether A is closer to B than C.” A number of such tests are performed and the results used to develop an overall picture of the subject’s representation. Estimation tasks avoid the errors associated with motor capabilities that are found in the previous method. However, researchers such as McNamara (Timothy P. McNamara et al., 1984) have argued that it is hard to properly interpret the results in such tests because the tests often conflate the
source of the phenomenon between the actual representation and the processes that act on it. Instead, he has suggested the use of spatial priming experiments where subjects are primed and their responses measured to create a more accurate picture of the representation itself. In this work, we only model experiments where the testing methods are map sketching and estimation tasks.

5.3 Requirements for Modeling Large-Scale Space

So far, we have described a bimodal version of the cognitive architecture Soar and shown how it interacts with an external diagrammatic representation. We now proceed to demonstrate that biSoar is suited for modeling the role of high-level cognition in representing and reasoning about large-scale space. One way to do achieve this is to model a few of the phenomena that have been discovered in experimental studies of large-scale space reasoning. However, because of the large number and variety of such phenomena, demonstrating biSoar’s ability to model one or a few of them is not a good indication of its suitability for the modeling the entire domain. Instead, we take a design approach and specify a list of suitability criteria that any agent architecture should satisfy if it is to be able to model the role of high-level cognition in large-scale space representation and reasoning.

There are three requirements that any proposal for modeling large-scale space should satisfy.

- Represent Information in symbolic & metrical Pieces.
- Intelligently combine them during problem solving
- Learn automatically during problem solving and show within and between task transfer of learned information.

The first criterion is based on established research in the cognitive architecture field. The second and third criteria arise naturally out of the interest in modeling using an architectural framework. I claim that the functionality present in the list is necessary, but there is no claim as to its sufficiency.
Represent Information in Symbolic and Metrical Pieces - The nature of the representation of large-scale space in humans has been a matter of contention. The term “Cognitive Map” that is most often used to refer to this representation, is considered to be misleading because it conveys the impression that the representation is similar to a real two dimensional map. Numerous experiments have shown that this is not the case and more appropriate terms such as “Cognitive Collages” and “Cognitive Atlas” have been suggested as alternatives. While the exact nature of the representation is still debated, there is consensus that spatial information is represented in both symbolic and metrical forms (Tversky, 1993). Moreover, there is strong evidence that this information is in pieces rather than as a single holistic representation (Tversky, 1993). Consequently, our first suitability criterion is that an architecture for modeling reasoning about large-scale space should accommodate both symbolic and metrical representations and that the represented information should be in pieces.

Intelligently Combine Pieces During Problem Solving - Given the piecemeal nature of the representation as required in the first criterion, it is obvious that the architecture should allow the agent to combine these chunks during problem solving. More importantly, the architecture should allow the agent to do so intelligently – i.e., during the process, the architecture should allow the agent to bring to bear all and only information that is relevant to the task under consideration. For example, in answering a question about wayfinding in Columbus, the agent should bring to bear information about temporary road closings in Columbus but not be overwhelmed by information about Cincinnati or Ann Arbor or other regions or states. This behavior ensures that the agent is able to efficiently solve problems in the presence of large amounts of information. McNamara lists Navigation, Wayfinding and Geographic Recall as the different types of large-scale space reasoning problems. Navigation, because it requires the agent to have local-space representations and associated processes that are, currently, not part of biSoar theory, is not a focus of this thesis. Thus, any proposal should be equally able to model both wayfinding and geographic recall problems.
**Learning** - The learning process should satisfy two conditions. One, it should be automatic; allowing the agent to learn as a matter of course during problem solving. Two, the agent should exhibit transfer of learning - both within a task and across different but related tasks. These two conditions are important only in the context of cognitive modeling. For a robot, learning can be the result of an explicit command and between-task transfer of learning is rarely a necessity. In modeling, however, subjects are rarely explicitly asked to learn a map or environment. Instead, they are more likely to be asked to solve a problem, and then tested to see what they have learned as a result of solving that problem. Moreover, it is frequently the case that subjects are tested on problems that are different from the one in which he/she was supposed to have learned the spatial representation.

\[
\begin{align*}
&\text{• If goal is find\_destination and agent at R2R5 and traveling Right on Route R2, then destination is P2, diagram is DRS1} \\
&\text{• If goal is find\_destination and agent at R1R3 and traveling Down on Route R1, then destination is R1R2, diagram is DRS2} \\
&\text{• If goal is find\_routes and at R1R4, then routes are R1, R4} \\
&\text{• If goal is find\_location and at R3R5 then diagram DRS3}
\end{align*}
\]

Figure 5.1: Examples of biSoar rules used in the wayfinding task

Together, these three criteria set up a basic threshold of suitability that any proposal should demonstrate in order to qualify for modeling cognitive map phenomena. It is likely that there are additional criteria, or that the existing criteria can be constrained further, but for our purposes, these three provide a sufficiently restrictive and useful set of conditions for modeling suitability.
5.4 Satisfying the Requirements in biSoar

5.4.1 Represent Information in both Symbolic and Metrical Pieces

In Chapter 2, we described how biSoar can represent diagrammatic information in addition to predicate-symbolic representations. This allows biSoar to satisfy one part of the first criterion – represent both symbolic and metric information. I now show how biSoar’s representations (in LTM) can be piece-meal in nature, thereby satisfying the second part of the first criterion. Recall that Soar’s LTM is in the form of rules. Each rule can be thought of as an if-then statement where the condition (if) part of the rule matches against the existing conditions in WM and the action (then) part of the rule describes the changes to be made to WM. Thus, Soar’s LTM is arranged to respond to situations (goals) that arise as part of problem solving. This makes sense because, ideally, a majority of an agent’s rules are learned as part of problem solving and hence in response to a particular situation.

Figure 5.2: A map showing some of the main routes in Columbus
If the situation (or a similar one) arises in the future, Soar can now use this learned rule in response. Since Soar’s rules are learned in response to problem solving situations, they are often specific to the goal and it is only through the use of many such rules that biSoar manages to solve larger goals. Fig 5.1 shows some examples of wayfinding rules (written out in English) in biSoar. Using such rules as well as others about location information, a biSoar agent can recall a map of an area.

5.4.2 Intelligently Combine Pieces During Problem Solving

To show how a biSoar agent can combine pieces of information to solve a spatial reasoning problem, I use an example wayfinding task based on the map shown in Fig 5.2. The agent has knowledge about various locations on the map and knowledge about routes between these locations. Fig 5.2 (b) gives a complete listing of all the knowledge that the agent has. The agent’s knowledge of the area is in the form of procedural knowledge and consists of bimodal rules (symbolic and diagrammatic components) in the following form:

If goal is find_destination and at location Lx and traveling in direction Dx on route Rx, then destination is location Ly, diagram is DRSx

Where DRSx represents the spatial extent of the section of the route that runs from Lx to Ly along route Rx. So, for example, the relationship between R2R5 and P2 in 5.2 would be expressed as

If goal is find_destination and at R2R5 and traveling Right on Route R2, then destination is P2, diagram is DRS1

The directions available are Right, Left, Down and Up (or East, West, South and North) though this choice of directions is arbitrary. More fine grained distinctions can be made if
needed. Along with the rule, there is also a diagrammatic object that is a simplified form of the relevant section of route Rx. For convenience, the above information is represented in a more concise form in 5.3 as follows:

\[ G_x, L_x, D_x, R_x \rightarrow L_y, D R S_x \]

<table>
<thead>
<tr>
<th>Gdest, R1R2, Right, R2 → R2R4</th>
<th>R1R2</th>
<th>R2R4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gdest, R2R4, Right, R2 → R2R5</td>
<td>R2R4</td>
<td>R2R5</td>
</tr>
<tr>
<td>Gdest, R2R5, Right, R2 → P2</td>
<td>R2R5</td>
<td>P2</td>
</tr>
<tr>
<td>Gdest, P2, Right, R2 → R1R2-1</td>
<td>P2</td>
<td>R1R2-1</td>
</tr>
<tr>
<td>Gdest, R1R2, Down, R1 → P4</td>
<td>R1R2</td>
<td>P4</td>
</tr>
<tr>
<td>Gdest, P4, Down, R1 → R1R4</td>
<td>R1R2</td>
<td>P4</td>
</tr>
<tr>
<td>Gdest, R1R4, Right, R1 → P3</td>
<td>R1R4</td>
<td>P3</td>
</tr>
<tr>
<td>Gdest, P3, Right, R1 → R1R2-1</td>
<td>P3</td>
<td>R1R2-1</td>
</tr>
<tr>
<td>Gdest, R1R4, Up, R4 → R2R4</td>
<td>R1R4</td>
<td>R2R4</td>
</tr>
</tbody>
</table>

Figure 5.3: Information in agent’s LTM for the wayfinding task
As we show below, during problem solving, the agent creates a map of the area, and finds a path from P4 to P2 that takes advantage of the spatial representation. Fig 5.4 gives the route-finding strategy used by the agent and 5.5 shows the route found by the agent.

1. Create a map of the area from available info if no external map present
2. Locate the starting & destination locations in the map
3. Make the starting location the current location
4. Find the routes on which the current location lies
5. For each route, find the directions of travel
6. For each route and direction of travel, find the next location
7. Calculate the Euclidean distance between these new locations and the destinations
8. Pick the location that is closest to the destination and make that the current point
9. Repeat 3-7 until destination is reached

Figure 5.4: The wayfinding strategy used by the biSoar agent

Figure 5.5: Route found by the biSoar agent from P4 to P2
5.4.3 Learning

The learning criteria constrains the space of models to those that can account for various phenomena involving learning in spatial reasoning. I demonstrate biSoar’s learning capability by showing transfer of learning both within a particular task and between related tasks.

5.4.3.1 Within-Task Transfer

Recall that biSoar’s learning capabilities are just manifestations of Soar’s chunking mechanism. Chunking is the impasse-driven automatic learning mechanism in Soar where the Soar architecture compiles knowledge used in solving an impasse (subgoal) into the form of a rule. This rule is then used whenever the agent is faced with a similar impasse. The *if* part of the learned rule or chunk consists of the WM elements that caused the impasse, as well as any elements in WM that were relevant to solving the subgoal. The *then* part consists of the actions that were made in the subgoal that led to the resolution of the impasse. For example, if subgoal was to find the destination given a location, route and direction, then the rule would look as follows

*If Goal is find_destination location is Lx and Route Rx and direction Dx then destination is Ly, Route Diagram DRSx*

Where DRSx is the diagrammatic component describing the section of the route Rx from Lx to Ly. For within-task transfer, we use a wayfinding scenario where the agent is given two wayfinding tasks between different locations on an external map. The routes between these pairs of locations have some common subset, i.e., there is a section of the route that is found in solutions to both these tasks. The third task is another wayfinding task between two locations, one that lies along the path from task1 and the other that is from the path in task2. The agent has to perform the third task without an external map, using only information that was learned from the previous two tasks. The tasks are based on the map (based on Columbus, OH) shown in Fig 5.2. Task 1 consists of finding a route between locations P1 and P2. The wayfinding strategy used is the same one shown in 5.4.
Fig 5.6 gives a partial list of the knowledge learned by the agent while solving the task. Task 2 consists of finding a route between locations P4 and R3R5. Fig 5.7 gives a partial list of the knowledge learned by the agent in solving this task.

<table>
<thead>
<tr>
<th>Gon,P1 → R1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Gon,R1R3 → R1,R3</td>
<td></td>
</tr>
<tr>
<td>Gon,R1R2 → R1,R2</td>
<td></td>
</tr>
<tr>
<td>Gon,R2R4 → R2,R4</td>
<td></td>
</tr>
<tr>
<td>Gon,R2R5 → R2,R5</td>
<td></td>
</tr>
<tr>
<td>Gdir,P1,R1 → up,down</td>
<td></td>
</tr>
<tr>
<td>Gdir,R1R3,R1 → up,down</td>
<td></td>
</tr>
<tr>
<td>Gdir,R1R2,R2 → right,left</td>
<td></td>
</tr>
<tr>
<td>Gdir,R2R4,R2 → right,left</td>
<td></td>
</tr>
<tr>
<td>Gdir,R2R5,R2 → right,left</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gdest,P1,R1,down → R1R3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1R3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gdest,R1R3,R1,down → R1R2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1R2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gdest,R1R2,R2,right → R2R4</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1R2 → R2R4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gdest,R2R4,R2,right → R2R5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R2R4 → R2R5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gdest,R2R5,R2,right → P2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R2R5 → P2</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.6: Knowledge learned by the agent in solving task 1
<table>
<thead>
<tr>
<th>GON, P4 → R1</th>
<th>GON, R1R4 → R1, R4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gdir, P4, R1 → left, right</td>
<td>Gdir, R1R4, R4 → up</td>
</tr>
<tr>
<td>Gdir, R2R5, R5 → up</td>
<td>Gdest, P4, R1, right → R1R4</td>
</tr>
<tr>
<td>Gdest, R1R4, up → R2R4</td>
<td>Gdest, R2R5, up → R3R5</td>
</tr>
</tbody>
</table>

Figure 5.7: Knowledge learned by agent in solving task 2

Figure 5.8: Map created by the agent for the geographic recall task
Task 3 is to find a route from R1R4 to P2 without the external map. The map created by the agent is shown in Fig 5.8 and the route found is shown in Fig 5.9. The agent uses the same wayfinding strategy (Fig 5.4) as used in tasks 1 and 2.

5.4.3.2 Between-Task Transfer Task
The between-task transfer example uses the initial wayfinding tasks as in the within-task transfer example, but the third task is switched to a geographic recall task where the agent is asked to recall the spatial relationship between a pair of locations on the map. The strategy used by the agent for task 3 is simple. Recall the map of the area as in the wayfinding task and simply retrieve the spatial relationship between the locations. The particular recall task is to determine if location P1 is north of location R3R5. Fig 5.8 shows the map created by the agent during problem solving.

5.5 Related Work
Spatial representations and spatial information processing play an important role in a number of cognitive activities – from high-level reasoning tasks such as wayfinding and geographic recall to low-level motor tasks such as object manipulation and object avoidance. Explaining how spatial reasoning takes place thus involves accounting for a wide range of phenomena. Currently, there are no theories that explain the entire range of spatial information processing phenomena. Researchers have traditionally concentrated on studying a subset of spatial phenomena, often dividing the problem with respect to the
various spatial tasks that are performed. For example, navigation, wayfinding, spatial memory etc. One such sub-area that is of interest is the role of high-level cognition in spatial reasoning tasks. biSoar is based on a theory of high-level cognition in which the role of spatial representations is particularly important. We have already compared biSoar with the spatial information processing abilities of two other theories of high-level cognition – Soar-VI and ACT-R/S – in a previous section.

In addition, because we illustrate the usefulness of biSoar for building models of wayfinding and geographic recall, it would be useful to look at other, more task-specific proposals, that focus on similar tasks. As it happens, there is a body of work that lies at the intersection of AI and Cognitive Science, developed for robot navigation but inspired by psychological and cognitive studies of human spatial reasoning, that can be usefully compared against biSoar’s representations and processes.

5.5.1 Task-Specific Theories
The task-specific theories that we compare biSoar against are Spatial Semantic Hierarchy (SSH) and Prototypes, Location and Associative Networks (PLAN). SSH and PLAN’s concern with spatial reasoning arises from their interest in robot navigation; specifically, they have the goal of explaining how a robot can learn the spatial aspects of its environment and use this learned representation for the purposes of navigation. This concern with learning leads to concern with representation, since the learning process needs a target representation. Spatial knowledge required for navigation is often a mixture of metrical, topological and symbolic (Criteria 1 above). If the environment has enough recognizable unchanging landmarks, a topological representation that gives information about the links between adjacent landmarks would suffice for many navigation tasks. Symbolic information such as “to the right of” may be additionally useful. In many cases, however, metrical information may also be called for, e.g., in following directions, navigation may call for traveling a certain number of miles in some direction. Another necessity in robot navigation is for the robot to recognize its presence at and orientation to various landmarks – this calls for some way to associate perceptual cues with the different landmarks. Our work on biSoar does not address the issue of
learning these representations from the environment, but we review the target representations learned by SSH and PLAN, and compare them to biSoar’s proposal for representing spatial information.

**Spatial Semantic Hierarchy** - The SSH (Kuipers, 2000) is a multi-layered theory that represents its knowledge of space at multiple levels – control, causal, topological and metrical, with the information at one level building on what was learned at the next lower level (except in the case of the metrical level, which utilizes information from both topological and control levels.) The control level represents distinctive states. A distinctive state, such as an intersection in the environment, is composed of sensor values that satisfy some applicability criteria such as a maximum on a control law like hill-climbing. The creation of such representations is outside the purview of biSoar because they are part of the perceptual/motor mechanism rather than high-level cognition. However, once created, the knowledge required for recognizing a distinctive state may or may not be stored in LTM. In some cases, recognition is done without the aid of high-level cognition, as in the case of everyday face recognition. In other cases, high-level cognition is an active participant in the recognition process. For example, if a distinctive state is described using a set of cues, such as “You are at state s1 if there is a wall to your left, a lamp straight ahead and a mirror to your right”, or, in the case of SSH, simply a set of sensor values, then, this cue information can be stored in the biSoar agent’s LTM and high-level cognition is involved in matching these cues during state recognition. In this case, the actual recognition processes that identify the wall, lamp and mirror might take place by automated perceptual processes without the involvement of high-level cognition, but once the recognition is performed, the association of the existence of these cues with the distinctive state is done by high-level cognition. Since SSH is not concerned with the above distinction, it does not distinguish between the two types of representations or say which one is preferred in SSH. If it is the first scenario, then it is outside of biSoar. If it is the second scenario, then biSoar can accommodate such representations and play a role in the recognition of the distinctive state.
At the causal level, distinctive states are linked together by the actions (control laws) that can be performed to get from one distinctive state to another. Most of the information at the causal level can be represented in biSoar. For example, the information that “If the agent is at state s1 and the destination is state s2 then the agent should keep going straight ahead” can be represented as a rule in biSoar’s LTM. Such a rule can also be learned by the biSoar agent as it traverses a local area. The topological level represents information in terms of places, paths and regions. Multiple states at the causal level can correspond to the same place at the topological level. As an example of this, consider an intersection. When the robot is facing North, it has a particular view, let’s call it v. If the robot now turns to the right, it has a new view, v’. At the causal level, these views are distinct, and along with the knowledge that the action of “turning right” from v will result in v’, forms the representation at that level. At the topological level in SSH, v and v’ are the same place. A place is different from another place only if some action that moves the robot is present. In a biSoar agent, there is no separate representation for topological information. LTM rules represent this information too, simply by referring to places and paths rather than actions and views. For instance, “if the agent is at location l1 and the destination is location l2 then go North on route r1” is an example of topological information in biSoar.

The metrical level can represent local places using a two-dimensional occupancy grid and, a spatial analog of the large-scale space that the agent has explored. Both these functionalities are natural to biSoar. The DRS can be used to represent both local spaces, as well as a map of the large-scale space. While SSH proposes a way in which 2-D metric information may be represented and used, biSoar, in particular, DRS provides a concrete representational format for metrical information. Further, the metrical representation in SSH is hard to create, forming only when the agent is extremely familiar with the environment, and used rarely even then. A biSoar agent, on the other hand, creates, modifies and inspects the information in DRS during problem solving, making it an integral part of the problem solving process.
SSH theory provides an architectural framework in the form of a hierarchy where different types of information are represented at different levels. In biSoar, there is only a single level (the LTM representations), but it provides the same functionality that SSH provides for the tasks we have discussed.

**Prototypes, Locations, and Associate Networks (PLAN)** - Though PLAN (E. et al., 1995) is proposed as a theory of the human spatial representation, the authors proceed with the understanding that the essential facility supported by such a representation is navigation and wayfinding. PLAN has 4 components – Landmark Detection, Path Selection, Direction Selection and Abstract Environment Overviews. Under PLAN theory, an agent in the world first identifies certain objects (either due to perceptual or functional salience) as important (landmarks). Like in the SSH case, this detection is outside the purview of biSoar but detected landmarks can be represented in biSoar’s LTM.

As the agent moves through the world, it starts to accumulate addition information such as how the various landmarks are connected. This process leads to the formation of a topological map. In biSoar this information is represented as LTM rules. As the agent spends more time in the world, it is able to identify not just landmarks but particular scenes and connect them together using various directional information. PLAN, captures a local view using a combination of grids and pie structures. Grids correspond to diagrams in biSoar and represents the current scene that is viewed by the agent. Each grid is divided in to a number of sub-regions such that each sub-region corresponds to some object within the scene. The objects are stored only in enough detail to separate them from other objects in that same. Locating objects within the grid are done via proprioceptive information from eye movements. When the agent is at a particular place, it can also move its head and cover an area that is larger than that covered by a single grid. This larger area is constructed from distinct grids that can be arranged in the form of a pie. Each slice of the pie corresponds to a grid. The pie is divided into 5 slices corresponding to five head directions – straight ahead, the two sides and the two 45 degree angles from straight ahead. This pie structure together with the individual grids
constitute the local map in PLAN. The locations information for the pie slice directions is said to be from proprioceptive information too – this time coming from the orientation of the head.

Grids in PLAN are similar to diagrams in biSoar, though the DRS representation is much more developed. Also, location information in PLAN and biSoar have similar sources of origin, proprioceptive memory. However, the role of proprioceptive memory is more sophisticated in the biSoar story. First, it corresponds to multiple locations, not just 5 slices, and secondly, it is imprecise, leading to the agent having to use other information to locate diagrams correctly. Finally, agents are able to produce an overall, survey type, map of the area being investigated. PLAN also calls for a more involved role for metric representations through the use of local maps (SSH “views” are, in some ways, similar to local maps). Local maps are diagrammatic representations of scenes at choice points, such as a fork in the road or a doorway. With experience, a route can be learned as a series of such local maps interspersed with direction changes and motion. Thus route information is stored at levels - between landmarks or between various scenes. This distinction is present as a result of the content of the various rules and as explained earlier, emerges, as a result of problem solving.

Like SSH, and unlike biSoar, PLAN too underutilizes its metric representation by not accounting for the creation and manipulation of diagrammatic objects as part of spatial problem solving.
The previous chapters have described the biSoar architecture and an interface and external representation for a biSoar agent that interacts with map-like sources. We have also evaluated biSoar’s abilities in modeling problem solving involving large-scale space by showing how it fulfills a set of suitability criteria. We have also described the advantages of modeling in biSoar. In this chapter, we describe example biSoar models of three different phenomena – simplification in recalled curves, distortions in geographic recall and influence of the spatial goal. In the case of simplification and geographic recall, we build multiple agents, each embodying a different explanation for the phenomena. We then show how the type of explanation of each model can aid in the design of as well as suggest control variables for additional human experiments. For the third phenomenon, we build a single model that explores one possible source of the phenomenon. There is no claim that the models we provide are the correct models of the phenomena. Instead, these models are examples designed to show biSoar’s versatility and flexibility in helping build models of large-scale space reasoning.
6.1 Task 1 – Simplification in Recalled Curves

Curves recalled from spatial memories, whether they are rivers in Paris or routes by cab drivers rarely preserve their exact curvature or their orientation to each other and to other landmarks (Tversky, 2000). Details in a curve such as the actual angles at intersections are lost and route curvature is usually straightened. We refer to this phenomenon as simplification and explore how simplification can arise from the architectural features of biSoar. In particular, we explore whether certain attention related features of the perceptual mechanism along with chunking (represent only that to which attention was paid) is enough to explain the emergence of simplification in recalled maps. We create two models of the simplification task and show how biSoar can be used to model different strategies to achieve the same goal.

Figure 6.1: Routes found by Simpl1 from the map in Fig 4.2(a)
(a) R1R4 to R1R5. (b) P4 to R1R3-1. (c) P1 to P2
6.1.1 Model 1
The agent (referred to as Simp1) is the same wayfinding agent described in the previous chapter. It is given the task of finding various routes in the map shown in Fig 6.2(a). Fig 6.1 shows the result of route-finding between a few locations on the map. The route-finding strategy used is the same as the one in the last chapter (Fig 6.2(c)). The critical step in the strategy is the step where, once the next point has been selected, the agent attends to the route from the current point to the selected next point. The resulting location and detail simplification via the interface results in a representation of the route that is simplified, according to the attentional demands of the task, in WM. When Soar’s chunking mechanism learns from the resolution of the sub-goal, it learns this simplified representation.

6.1.2 Model 2
A new agent (Simp2) is created for the same task. Simp2’s strategy is the same as Simp1’s except that Simp2’s problem solving strategy involves attending to only the locations of various intersection points as well as the symbolic information about which
routes to take between the intersection points. During recall, Simp2 recalls these locations and symbolic information about connecting routes and constructs the path between the starting and destination locations. Fig 6.2 shows the various routes constructed by the agent.

The two models (represented by the two agents Simp1 and Simp2) indicate two different explanations for the simplification phenomenon. The simplified routes recalled by Simp1 are the result of architectural features of biSoar – loss of detail via the interface and bimodal chunking. Depending on which aspects of the routes that attention was paid to, Simp1 chunks a simplified version of the original route. Simp2 on the other hand, does not even bother trying to chunk the curves. Instead, it learns the locations of important intersections and the routes they are on and connects the intersections with straight lines during recall. As mentioned earlier, the ability to create these models does not automatically suggest that either explanation is the definitive source of the simplification phenomenon. There could be other as yet unwritten models that might turn out to be, in fact, right. However, these models do suggest that one variable to control for is whether subjects are recalling only locations or both locations and routes. One way to do this would be to have a particularly attention grabbing feature on one of the curves (maybe a loop or sudden change in direction). Each model also allows the experimenter to develop further tests. Since Simp1 is an architectural explanation, the phenomenon should be found across all agents. On the other hand, the explanation provided by Simp2 is due to a particular strategy and it should be possible to influence how subjects solve the route problems and affect the phenomenon.

6.2 Task 2 – Distortions in Geographic Recall

According to Stevens and Coupe (Stevens & Coupe, 1978), when subjects were asked about the relation between San-Diego and Reno, most answered that San-Diego was to the west of Reno even though in reality, Reno is west of San-Diego. Stevens and Coupe suggested that this result indicated two things – one, that the cognitive map was unlikely to be a faithful metrical representation and two, that the representation was hierarchical in nature, the hypothesis being that since the subjects did not have any information about
the relationship between San-Diego and Reno they went up the hierarchy and compared the containing regions – California and Nevada. Since California is to the West of Nevada, it followed that San-Diego was to the west of Reno.

In a hierarchical structure, relations common between a large number of subordinate nodes are stored as the relation between their respective parent nodes. This strategy allows for efficient storage, since redundant information can often be removed from lower levels and stored as relations between nodes higher up the hierarchy. However, it can also give rise to errors such as in the case of the SD-Reno example. They conducted further experiments, this time using simple maps that were created in the lab and given to subjects to study for a brief while. Their initial findings were duplicated and they found that subjects were likely to mistakenly report the relationship between points in the map as the same as the relationship between their containing regions. Hirtle & Jonides (Hirtle & Jonides, 1985) conducted a number of experiments that showed that hierarchical representations are formed even in the absence of explicit boundaries. That is, people naturally grouped locations into regions (either geographically or using other more functional criteria) and represented them in a hierarchical manner.

McNamara (T. P McNamara, 1986) conducted experiments to distinguish between nonhierarchical, strongly hierarchical and partially hierarchical representations.
for spatial memory. In a nonhierarchical representation, information about relations between objects is represented in the form of propositions with no hierarchical structure. In a strongly hierarchical representation, a hierarchy exists but relation information between different branches of the tree is not stored. Instead, such information has to be inferred from the relationship of nodes higher up in the tree. Finally, partially hierarchical representations are similar to strongly hierarchical representations but allow relation information between branches to be present. McNamara found, as a result of priming experiments, that subjects were encoding spatial relations across region boundaries which called for a partially hierarchical representation.

He also found that while he was able to replicate Stevens and Coupe’s findings about direction estimation errors, the distortions were more when the distances between the locations were shorter than if they were longer. He concluded that the evidence for partially hierarchical representations and the effect of distance on distortion were evidence that Stevens and Coupe’s explanation for the phenomenon – that the distortions were due to the lack of relation information between different branches of the hierarchy – was inadequate. Instead, he proposed that the distortion could be because higher order spatial information was being weighted more heavily during the inference process. This weighting was due to subject’s uncertainty about the location information of the cities. When the cities were close together (the distances between them were shorter), even slight inaccuracies would result in large error. If the cities were far apart (the distances between them were longer), slight inaccuracies would not result in large error. Thus, when the cities were close together, and location information was uncertain, there was a tendency to depend more on the spatial relation between higher order regions accounting for both the distortion and the distance effect on the distortion.
More recently, Barkowsky (Barkowsky, 2001) has provided a computational account of the San-Diego Reno example that uses a diagrammatic representation and reasoning system called MIRAGE. When finding spatial relations between entities, the system first accesses the appropriate information from its hierarchical spatial LTM via a bi-directional search that finds a path linking the entities. Fig 6.4 shows MIRAGE’s hierarchical LTM representation for the SD-Reno example. If there are multiple paths between the entities, other factors such as time and cost constraints are used to pick the path. Once a path has been selected, the information represented by the path is visualized as a diagrammatic representation. If enough information is not present in the path to go from the symbolic LTM representation to the diagrammatic visualization, additional, default information, such as the use of squares to represent regions are used. Fig 6.5 shows the visualized image in the SD-Reno example. Once the visualization is complete, MIRAGE simply reads off the relation between the entities.

Figure 6.4: The hierarchical LTM representation of the SD-Reno example in MIRAGE. (Barkowsky 2000)
We built three different models for this task. Model 1 is of an agent that has the explicit task of learning a map representation. It uses the following strategy for map learning:

The agent is given the task of learning the diagram shown in Fig 6.6(a). The agent’s strategy for learning a map is as follows –

- Learn overall simplified representation of the given map (Fig 6.6(b)) that concentrates on only the regions present.
- Learn the locations of individual cities by learning a simplified version of their containing regions and locating the city (Figs 6.6(c) and 6.6(d)).
- Finally, learn the detailed structure of the each region. Fig 6.6(e) the pieces learned for the California region.

The overall simplified representation learned in step 1 is a simplified view of the regions of the map. The current model learns the entire as a single simplified diagram but in the case of more complicated maps other factors such as the map size as well as an agent’s attentional demands will have to be factored in to decide what the learned representation will look like. Similarly, in step 2, the current model learns a single diagram for each region. Each diagram consists of the simplified region and all the points in that region, but as in the case of step 1, this need not always be true. Finally in step 3, the agent learns...
the detailed structure of each individual region in the external map. Due to loss of detail, the agent is forced to break each region down into smaller pieces and learn those pieces.

When the agent is given the task of recalling the locations between SD and Reno, the agent looks up San-Diego and finds that it is in California, looks up Reno and finds that it is in Nevada and combines California and Nevada into a single diagram and reads off the relationship from it. The model does not produce the distortion because even with the simplified diagrams, Reno still ends up East of San-Diego. The diagram generated by the model is shown in Fig 6.7(a).
Model 2 has symbolic information in LTM that San-Diego is South of San Francisco and that Reno is East of San Francisco. It constructs a diagram (Fig 6.7(b)) in WM using this information and extracts the (wrong) answer from the diagram. Model 2 produces the distortion phenomenon.

Model 3 has symbolic information in LTM that San Diego is in California, Reno in Nevada and that California is to the West of Nevada. This information is used to construct a diagram (Fig 6.7(c)) and the (wrong) answer extracted from it. Like model 2, model 3 also produces the distortion.

The variety of models in Task 2 exhibit biSoar’s flexibility in modeling spatial phenomena. Each model provides a different explanation and, in essence, suggests a separate control variable. Take, for example, model 2. The explanation suggested by the model is that subjects use a specific strategy – that of comparing the location of the target cities to a common city and inferring the relationship from that knowledge. This strategy
can be controlled for by using artificial maps (as Stevens and Coupe do in their original paper) that do not provide this extra information. Similarly, model 3 is a variation of the hierarchical model first suggested by Stevens and Coupe (Stevens & Coupe, 1978), where the control variable is the presence of containing regions in the map. Thus, models in biSoar have a straightforward mapping to issues to control for and building these models provides a natural way of discovering these issues. Of course, the experimenter is free to simply think of various explanations without modeling in biSoar, but the advantage is that it provides additional constraints and enables the experimenter to let biSoar generate the consequences of the assumptions. As mentioned earlier, we do not know of any systematic way of generating these models/variables but heuristics such as “look for at least one explanation from each possibility in the explanation space” can suggest lines along which the model builder/experimenter may approach the problem.

### 6.3 Task 3 – Modeling the Influence of Goals in Spatial Memory

Taylor, Naylor and Chechile (Taylor et al., 1999) have explored the influence on the cognitive map of the spatial perspective and spatial goals of the problem solving agent. In their experiments, they had subjects learn an unfamiliar campus environment by having them either solve a route goal such as wayfinding or a survey goal such as geographic recall (different spatial goals). In the case of either goal, some subjects were asked to solve the problem by navigation and others by using a map (different spatial perspectives). The subjects were then tested for effects of the difference in perspective and goals. The results suggested that there were multiple influences on the representation created including the learning condition (navigation vs map-reading) and the spatial goal (wayfinding vs geo recall). The main effect of the spatial goal was found in tasks where the subjects used a map.

In the particular set of experiments that showed effects of the spatial goal, subjects were divided into two groups. Each group was given the same map - one floor of the psychology building at Tufts University with the various rooms marked and labeled. The first group had a survey goal where they were given a list of rooms and asked to find the adjoining rooms between each room on the list and the rooms adjoining it. The other
group had a route goal where they were given 10 pairs of locations and asked to find the shortest routes between these locations. Each group was then given a variety of tasks that exercised the cognitive map representation that they had just learned. The main effect of the influence of the spatial goal was found in two tasks – the “walk-through-the-wall” task and the “map recall” task. In the “walk-through-the-wall” task, subjects were given a room name and a direction and asked to estimate which room they would end up in if they walked through the wall of the given room in the specified direction. Subjects who had trained on the survey goal were better at this task than those who trained using a route goal. A similar effect was found in the “map recall” task where the subjects were given only the outer walls of the building and asked to fill in the various room labels and boundaries.

We create two biSoar agents, one for the route-goal model and the other for the survey-goal model. In our setup, instead of a map of a building we use the map of Columbus first introduced in Fig 5.2 (reproduced below).

Figure 6.8: A map showing some of the main routes in Columbus
**Learning Phase** - As in the Taylor, Naylor & Chechile experiment, for the route goal, the agent has to find the shortest routes between 10 pairs of locations in the map. For the survey goal, since there are no adjoining locations in the route map, the agent is instead given pairs of locations and asked to learn the spatial relationship between them. The pairs are chosen by first selecting a location and then choosing other locations that are adjoining to it. So for example, one location chosen would be R2R4, and the pairs would be (R2R4,R1R4) (R2R4,R2R5)(R2R4,R1R4) and so on. Such a list simulates the adjoining rooms nature of the original experiment. The agent’s strategy in this task is to imagine straight lines connecting the pairs of points and learn those. Fig 6.9 gives a list of the pairs used in the route goal and Fig 6.10, the list of locations used in the survey goal.

**Testing Phase** – For the testing phase, each agent is given the task of recreating the map based only on the information it has learned in the learning phase. The route goal agent is given pairs of points to place on the map which it does by placing the starting point and

\[(P1,P2),(P4,R1R2-1),(P3,R2R5),(R1R2,R3R5),(R2R4,P2),(P4,P3),(R1R3,R1R4),
(R3R5,P3),(P1,R2R4),(P2,R3R5)\]

**Figure 6.9**: list of locations in the route goal between which the agent has to find the shortest routes

\[(P1,R1R3),(R1R3,R1R2),(P4,R1R4),(R2R4,R1R4),(R1R4,P3),(R2R4,R2R5), (R2R5,P2),(P2,R1R2-1),(R3R5,R2R5),(P4,R1R2)\]

**Figure 6.10**: list of adjoining locations in the survey goal between which the agent has to find the spatial relationship
finding a route from it to place the destination. If the starting point is already present it simply finds the route to the destination point and places that. If the destination point is already present it finds a route from the destination to the starting point and places the starting point. The survey goal agent is given pairs of adjacent locations to place on the map.

Recall that there is location inexactness associated with the locations attended to by biSoar in the external representation. Due to this inexactness, the agents create slightly different maps each time they are run. To average out such differences, we ran each agent one hundred times. We calculated the total error in a map as the sum of the Euclidean distances between the positions of locations in the original map and the positions of locations in the map created by the agent. The average total error over the hundred runs for each model was calculated. We found that for these particular models, the average total error for the survey-goal agent was less than that for the route-goal agent. Qualitatively, this error is similar to that found by Taylor, Naylor and Chechile. However, it is worth emphasizing the point that this example is only meant as a demonstration of how the effects of particular models – architectural or otherwise – can be tested in biSoar. There is no claim that because the above models reproduce the findings, they also correctly identify the cause underlying the phenomenon of goal influence. In fact, the strategy used by the route-goal agent in recreating the map is much too simple for it to be used as a model for human map recall. Similarly, the small changes in learning tasks may nevertheless have significant impact on the overall task. Models and tests have to be carefully designed before any conclusion about the nature of human performance can be drawn by using biSoar. The example merely shows that the biSoar framework is capable of supporting such models.
CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion
The overall goal of the work presented in this thesis is to investigate the design, implementation, and uses of a multi-modal cognitive architecture. The architecture that we presented, namely biSoar, represents an important first step in that direction. We took a traditional predicate-symbolic cognitive architecture and provided it with one additional modality – a visual modality restricted to diagrammatic reasoning – making the architecture bimodal. The restriction on the number of modalities as well as limiting the architecture to diagrammatic reasoning allowed us to investigate multi-modal issues without the additional burden of detailing extra modalities and solving important but unrelated vision problems. Specifically, building biSoar allowed us to study the problem solving process, the structure of short and long term memories and the changes to the learning mechanism in an agent with multiple modalities. We believe that our observations about how the two modalities interact are general enough to be extended to the situation with other perceptual modalities.

We then proceeded to investigate the usefulness of biSoar as a cognitive modeling tool. For this, we chose the domain of large-scale space reasoning, an area where the presence of a spatial representation is particularly helpful. We showed how biSoar can be
used to write agents that can solve various spatial problems such as wayfinding and geographic recall, and used to implement models of spatial phenomena. Modeling in biSoar provides some unique advantages over the traditional approach of building task-specific models and we elaborated on these advantages over the course of building models for three different large-scale space phenomena.

7.1.1 Contributions

The thesis provides a number of contributions about the nature of bimodal representations and reasoning as well as the application of such an architecture to cognitive modeling. Contributions include

- The bimodal architecture biSoar and a description of the nature of the bimodal cognitive state in a biSoar agent was given in Chapter 2. An existing diagrammatic representation system, DRS, was used to augment Soar’s symbolic representation such that the agent’s WM was made up of both symbolic and diagrammatic information. Additionally, the LTM of the agent was also bimodal, with rules matching to the symbolic part of WM and the actions modifying both symbolic and diagrammatic parts. An example in the blocks world domain was used to demonstrate how problem solving proceeds in this bimodal architecture.

- An account of chunking, Soar’s automatic learning mechanism, for the bimodal case was also presented in Chapter 2. Bimodal chunking extends Soar’s chunking such that all information that was relevant to solving the sub-goal, irrespective of whether it is symbolic or diagrammatic, is chunked. For the diagrammatic case, any diagrammatic object that was accessed or perceptual/action routine that was operated on the diagrammatic part of WM diagram, and is relevant to the solution becomes part of the chunk.

- Chapter 3 discussed the advantages of having a bimodal cognitive state in dealing with certain aspects of the Frame Problem. The differences between the approaches adopted by a traditional Soar agent and the bimodal biSoar agent were illustrated using examples from the blocks world domain. A quantitative analysis of the time and space demands of both agents was also provided.
• Chapter 5 introduced the domain of large-scale space reasoning. We presented a set of criteria – represent information in symbolic and diagrammatic pieces, intelligently combine them during problem solving, learn as a result of problem solving and show within- and between-task transfer of learned knowledge – that any system that models large-scale space should posses. We demonstrated how a biSoar agent satisfies the above criteria using problem solving examples involving wayfinding and geographic recall.

• Chapter 6 demonstrated biSoar’s modeling abilities. We took three different phenomena involving large-scale space reasoning – simplification of recalled curves, distortions in geographic recall and influence of the spatial goal - and built models in biSoar that exhibited these phenomena. These examples also served to highlight the advantages of using the architectural approach in cognitive modeling.

7.2 Future Work

While biSoar represents an advance over current theories of cognitive architecture, there are a number of issues that remain to be addressed. Issues can be thought of as roughly falling into one of thwo groups – Architectural issues and Modeling Issues. Architectural issues focus on improving and extending the biSoar architecture while modeling issues deal with improving biSoar’s modeling abilities.

Architectural Issues – Broadly, this category can be divided into structural/representational issues and issues about the role of these structures in the multi-modal process. One such structural issue is the nature of episodic memories. Episodic memories are unique in that they have perceptual content, and the presence of perceptual representations in biSoar make it a good candidate to explore how such memories are formed and used by an agent. This may require additional LTM representations in biSoar as the current rule-based LTM does not support memory retrieval based on partial-matches.
How episodic memories are used during problem solving in biSoar falls in the second category of architectural issues. Since an episode is likely to be a sequence of some combination of symbolic and perceptual information, only a part of which is, usually, required to solve the current problem, it is unclear how an agent will access the relevant part of each memory. Another such issue is the role of diagrammatic representations in procedural LTM. In Soar (and biSoar), the LHS of procedural LTM rules are matched to the contents of WM to decide which rules to execute. When LTM is symbolic, the matching process is straightforward. If it is bimodal and diagrams were present in the LHS of rules, we would need some way of matching a diagram in LTM to (a part of) the diagram in WM. In our current implementation, we side-step this issue because none of our examples have a need for such matching. It is unclear whether such matching is necessary but a complete theory of multi-modal cognitive architectures requires an account of the role of perceptual representations in procedural LTM. A third area of future work is the interaction between DRS elements from perception and imagination. Such interplay is important in certain tasks that involve information from both sources such as object comparisons in Pinker’s graph problems (Pinker, 1990) (Trafton et al., 2002).

Both types of problems that are detailed above extend beyond diagrammatic representations to other, as yet unexplored, perceptual modalities. In fact, it is not even clear at this point how many such perceptual modalities there are. Furthermore, additional modalities may also have their own unique representational issues such as the case with the kinesthetic modality. In Chapter 3, we augmented biSoar such that it has access to location information. Our hypothesis is that such information is from proprioceptive sources and possibly is part of the kinesthetic modality. While we provided biSoar with limited capabilities in this respect, there are many issues remaining about how it is represented by the architecture and subsequently used during problem solving.

**Modeling Issues** – Currently, biSoar only models spatial memory phenomena. That is, biSoar models can be built to understand the effects of architectural, strategic and knowledge commitments on the kinds of representations that are formed while solving a
problem. Other architectural theories, especially ACT-R, go beyond this capability and allow modelers to extract timing information from models that allow them to match the agent’s performance against human performance at the quantitative level. While biSoar retains some of this ability at the symbolic level from Soar, the DRS was designed with keeping purely functional requirements in mind. In order to allow timing information to be associated with diagrammatic operations, it is important to design the perceptual and action routines such that they are based on primitive routines that mimic the human perceptual system. For example, the human perceptual system takes longer to compute relations between objects that are farther away in the external representation while current perceptual routines take the same amount of time.

Another facet of making biSoar ready for modeling involves specifying values for many of the parameters used in the architecture including the degree and nature of detail loss and location inexactness. Also, it may be helpful to don the hat of an experimenter by stepping through the entire modeling process – from identifying the phenomenon of interest, to writing biSoar models to identify control variables and using that information to create and refine various experiments that can in turn be used on subjects. This process along with the goal of modeling various diverse phenomena may also be helpful in bringing forth various architectural issues.


