From Verbs to Tasks: An Integrated Account of Learning Tasks from Situated Interactive Instruction

by

Shiwali Mohan

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2015

Doctoral Committee:

Professor John E. Laird, Chair
Professor Edmund H. Durfee
Professor Richard L. Lewis
Associate Professor Edwin Olson
Associate Professor Andrea Lockerd Thomaz, Georgia Institute of Technology

Intelligence is what you use when you don't know what to do.

—Jean Piaget

To Ma

Acknowledgments

I would like to thank my advisor John Laird who has been an inspiring teacher as well as an approachable and caring mentor. He motivated me to explore my own ideas, even when my progress was slow in the initial years. When I was too shy to talk, he constantly encouraged me to contribute more in classes and meetings. As I grew as researcher, he gave me the freedom to develop my own agenda. In his pursuit of understanding human-level intelligence, in his desire to learn, in being critical of his own work, and in being humble, he has set an excellent example of a researcher for me to strive for.

I would also like to thank my committee members - Ed Durfee, Rick Lewis, Edwin Olson, and Andrea Thomaz without whose guidance this thesis would not be in its current shape. Their constructive feedback has raised the standard of my thesis. Ed's insistence on clearly identifying my claims and contributions has made the dissertation more readable. Rick's class on *psychology of language* helped me have an holistic view of the human language faculty and inspired me to pursue questions related to language understanding and learning. Without Edwin, Rosie would have never been able to live a happy life in the real-world. Andrea's research and guidance has helped me keep my work grounded in HRI challenges.

Past and current members of the Soar lab have greatly influenced my research. Thanks to Nate Derbinsky for patiently listening to my half-baked ideas and guiding me in writing my first few papers; to Joseph Xu for constructive feedback on my papers and talks and for always having the time to help me debug my code; and to Justin Li for thoughtful discussions on various aspects of my thesis, for fixing Soar kernel bugs, and for always

being ready for a cup of coffee. This thesis would not have seen the light of day if Aaron Mininger and James Kirk hadn't put in tremendous effort in designing Rosie, writing code, fixing bugs, and in doing great research. I continue to be in awe of Aaron's engineering and James's hacking skills. I thank Steven Jones and Elizabeth Mamantov for their excitement and 'new-grad-student' optimism. It cheered up the cynic in me while I was writing this dissertation.

I thank Emily Mower Provost and Maya Cakmak for their encouragement and invaluable advice about the future. They have inspired me to continue as a researcher after I graduate. Thanks to my friends - Yash Adhia, Fatema Haque, Ramya Iyer, Rahul Jha, Vasudev Lal, Vineet Raichur, and Aayush Shah for making my time in Ann Arbor memorable. I am also thankful to Sulochana Dhar, Parul Jain, and Parul Batra for being my cheering squad all through college and grad school.

I am grateful for the continuing love and support of my family. My sister Monali Mohan and my brother Ravi Mohan always had the time to talk to me about things that were fun and things that were not. Without my mother Veena Gupta, I would not have had the courage to come to the US and pursue graduate studies. I have relied on her for heartening conversations whenever I felt discouraged about hard classes, rejected papers, and the relentless Michigan winters.

Thanks to my loving husband Shekhar Mittal for being by my side and helping me through my darkest moments and for being incredibly excited about my work.

Contents

Dedication . . . ii

Acknowledgments . . . iii

List of Tables . . . viii

List of Figures . . . ix

Abstract . . . xi

Chapter 1 Introduction . . . 1
- 1.1 Research approach . . . 5
  - 1.1.1 An Integrated Account . . . 5
  - 1.1.2 Architectural Implementation . . . 6
  - 1.1.3 Qualitative and Functional Evaluation . . . 6
- 1.2 Looking Ahead . . . 7

Chapter 2 Situated Interactive Instruction . . . 9
- 2.1 Situated Interactive Instruction . . . 9
- 2.2 Properties . . . 10
  - 2.2.1 Modality . . . 10
  - 2.2.2 Control . . . 12
  - 2.2.3 Content . . . 13
- 2.3 Desiderata . . . 15
  - 2.3.1 Situated Comprehension . . . 15
  - 2.3.2 Integrative Interaction . . . 16
  - 2.3.3 Incremental Learning . . . 17

Chapter 3 Related Work . . . 19
- 3.1 Grounded Language Semantics . . . 19
  - 3.1.1 Computational Linguistics . . . 19
  - 3.1.2 Integrated AI Systems . . . 21
  - 3.1.3 Robotics . . . 22
- 3.2 Human-Agent Dialog . . . 22

3.2.1 Virtual Agents . . . 23
3.2.2 Conversational Robots . . . 23
3.3 Interactive Learning . . . 24
3.3.1 Learning from Low-level Interactions . . . 24
3.3.2 Learning from High-level Interactions . . . 25
3.4 Summary and Discussion . . . 28

Chapter 4 Rosie Overview . . . 29
4.1 The Soar Cognitive Architecture . . . 29
4.1.1 Architecture Overview . . . 29
4.1.2 Analysis . . . 32
4.2 Rosie . . . 33
4.2.1 Environment . . . 33
4.2.2 Knowledge Representation . . . 35
4.2.3 Interaction Cycle . . . 39
4.3 Summary and Discussion . . . 42

Chapter 5 Verbs and Tasks . . . 44
5.1 Prior Work on Grounded Verb Semantics . . . 46
5.2 An Analysis of Task Verbs . . . 47
5.2.1 Chores Dataset . . . 48
5.2.2 Semantic Analysis . . . 48
5.2.3 Task-Oriented Analysis . . . 54
5.3 A Task-Oriented Representation for Grounded Verb Semantics . . . 56
5.3.1 Syntactic Knowledge . . . 58
5.3.2 Association Knowledge: Map . . . 59
5.3.3 Structural Knowledge . . . 59
5.3.4 Procedural Knowledge . . . 61
5.4 Discussion . . . 64
5.5 Looking Ahead . . . 65

Chapter 6 Comprehending Situated Commands . . . 67
6.1 Comprehending Imperative Sentences . . . 68
6.2 The Indexical Hypothesis . . . 70
6.3 Our Approach . . . 71
6.4 The Indexical Model of Comprehension . . . 73
6.4.1 Indexing . . . 75
6.4.2 Instantiating Domain Knowledge . . . 77
6.4.3 Meshing . . . 78
6.4.4 Active and Expandable . . . 79
6.5 Reference Resolution . . . 81
6.5.1 Non-linguistic Contexts . . . 82
6.5.2 Resolving References in the Indexical Model . . . 84
6.5.3 Integrative Processing . . . 86
6.6 Unexpressed Argument Alternations of Verbs . . . 92

6.6.1 Exploiting the Hearer's Instructional Experience . . . 92
6.6.2 Integrative Processing . . . 94
6.7 Summary and Discussion . . . 97

Chapter 7 Maintaining Flexible Dialog . . . 100
7.1 Collaborative Discourse Theory . . . 101
7.2 Our Approach . . . 102
7.3 Mixed-Initiative Dialog Model . . . 103
7.3.1 Interaction State Representation . . . 103
7.3.2 Interaction Management . . . 106
7.4 Evaluation . . . 107
7.4.1 Adaptive, Flexible Dialog . . . 109
7.4.2 Mixed-Control of Learning . . . 111
7.5 Summary and Discussion . . . 114

Chapter 8 Learning Goal-Oriented Tasks . . . 116
8.1 Explanation-based Generalization . . . 117
8.2 Our Approach . . . 118
8.3 Formulating Task Learning as EBG . . . 120
8.3.1 Assumptions . . . 120
8.3.2 Formulation . . . 120
8.4 Interactive Task Learning . . . 121
8.4.1 Executing a Known Task . . . 122
8.4.2 Learning a New Task . . . 125
8.4.3 Learn Associative Default Values . . . 134
8.5 Evaluation . . . 141
8.5.1 Comprehensiveness . . . 141
8.5.2 Generality . . . 143
8.5.3 Transfer . . . 147
8.5.4 Mixed-Initiative . . . 150
8.6 Summary and Discussion . . . 152

Chapter 9 Conclusions . . . 154
9.1 Future Work . . . 157
9.2 Conclusion . . . 159

Appendix . . . 161

Bibliography . . . 163

List of Tables

Table

5.1 Thematic roles represented in the chores dataset 50
5.2 Syntactic frames represented in the chores dataset 53

8.1 Learned tasks, parameters, policy space, and goals 141

List of Figures

Figure

4.1 The Soar Cognitive Architecture (adapted from J. Laird, 2012) 30
4.2 Rosie's table-top workspace and the interaction window. 34
4.3 Rosie Structural Diagram 36
4.4 Rosie's Interaction Cycle 40
4.5 Rosie's learning interaction trace. 40

5.1 A task-oriented representation for grounded verb semantics in Rosie's memories. Nodes and symbols in color purple correspond to spatial knowledge, orange to perceptual knowledge, green to goal definition and constraints, blue to procedural knowledge, and red to syntactical knowledge. . 57

6.1 Environment state and the knowledge encoded in Rosie's semantic memory. The white nodes (`P4, L1, M2, L2, L3`) represent indexical maps between amodal linguistic symbols (in red: `right, red, move`) and modal domain knowledge. Yellow nodes (`R10, R11, R12, A31`) represent spatial symbols and slots (round rectangles: `A31`), blue nodes (`V1, A11, A21`) represent visual symbol and slots, and green nodes (`P2, P3, G2, G3`) represent procedural symbols. 74
6.2 Number of agent-initiated interactions per task command. The boxes show task commands 1, 15, and 25. The words in color blue are unknown to Rosie when that task command was given. 81
6.3 Number of object queries asked by Rosie for RE resolution. 90
6.4 Number of interactions required for comprehending verbs in different alternations. 96

7.1 Annotated human-agent dialog for acquisition of *store* 104
7.2 Number of agent-initiated interactions per task command in different initial stages of prior knowledge. 110
7.3 Initial and final environment states for learning *set the table task*. 111
7.4 Human-agent interaction trace for instructor-controlled teaching strategy (simplified). 112
7.5 Human-agent interaction trace for agent-controlled teaching strategy (simplified). 113

8.1 Semantic representation of *store*. . . . 123
8.2 (left) Interactions for learning *store* with exploration depth $K = 0$. (right) Interactions for learning *store* with exploration depth $K = 2$. . . . 125
8.3 Simplified Soar trace for interactive task execution. . . . 126
8.4 Simplified Soar trace for retrospective instruction-aided learning. . . . 132
8.5 Associative default values for *store*. . . . 136
8.6 Relaxed associative default values for R3 and A4 in structural knowledge of *store*. . . . 139
8.7 New associative default values for R3 and A4 in structural knowledge of *store*. . . . 140
8.8 Learned hierarchical policies. . . . 142
8.9 Procedural generalization during task learning. . . . 145
8.10 Conceptual generalization for learning implicit parameters of tasks. . . . 148
8.11 Instruction assisted transfer during learning. . . . 149
8.12 Learning *store* at different depths of exploration. . . . 151

Abstract

Intelligent collaborative agents are becoming common in the human society. From virtual assistants such as *Siri* and *Google Now* to assistive robots, they contribute to human activities in a variety of ways. As they become more pervasive, the challenge of customizing them to a variety of environments and tasks becomes critical. It is infeasible for engineers to program them for each individual use. Our research aims at building interactive robots and agents that adapt to new environments autonomously by interacting with human users using natural modalities.

This dissertation studies the problem of learning novel tasks from human-agent dialog. We propose a novel approach for interactive task learning, *situated interactive instruction* (SII), and investigate approaches to three computational challenges that arise in designing SII agents: *situated comprehension*, *mixed-initiative interaction*, and *interactive task learning*. We propose a novel mixed-modality grounded representation for task verbs which encompasses their lexical, semantic, and task-oriented aspects. This representation is useful in situated comprehension and can be learned through human-agent interactions. We introduce the Indexical Model of comprehension that can exploit extra-linguistic contexts for resolving semantic ambiguities in situated comprehension of task commands. The Indexical model is integrated with a mixed-initiative interaction model that facilitates a flexible task-oriented human-agent dialog. This dialog serves as the basis of interactive task learning. We propose an interactive variation of explanation-based learning that can acquire the proposed representation. We demonstrate that our learning paradigm is efficient, can transfer knowledge between structurally similar tasks, integrates agent-driven

exploration with instructional learning, and can acquire several tasks. The methods proposed in this thesis are integrated in Rosie - a generally instructable agent developed in the Soar cognitive architecture and embodied on a table-top robot.

Chapter 1
Introduction

With the recent advances in artificial intelligence, computational agents have begun to take on new roles as intelligent collaborators in the human society. Virtual assistants - *Siri* and *Google Now* - are on their way to becoming standard phone interfaces, providing human users a novel way of interacting with their cellphones and accessing information. It is expected that general-purpose, personal robots will become pervasive in domestic, public, and industrial spaces within the next decade. They will assist humans in a variety of activities including doing household tasks and collaborating on the assembly line. Personal robots, along with other intelligent agents such as smart homes and cars, will add tremendously to the quality of human life. They will offer persons with impairments more independence, help older adults with their daily chores, transport people and goods, and perform search and rescue in environments that are too dangerous for humans.

Several challenges have to be addressed to make progress towards this vision. Each home, office, or assembly line is organized differently. Users will want the agents to perform a variety of tasks and will have different preferences. Customizing every agent for its deployment environment and user preferences is resource intensive and costly. One approach to this challenge is designing a generally intelligent agent that can adapt to the user requirement on its own instead of relying on dedicated programming. This approach requires that the agent be an efficient online learner. It needs to learn object recognition, semantic organization and categorization, spatial relations, and tasks from experiences in its environment and exploit this knowledge immediately for performance. Learning

through self-directed exploration alone can be challenging and slow, requiring numerous interactions with the environment. This has motivated research on human-agent interaction driven learning paradigms that studies how human feedback, guidance, and structure can be used to reduce the complexity of learning. The goal is to develop agents that can be easily extended by human users using natural interactions to perform new tasks.

Recently, interactive task learning (ITL) was identified as a challenge problem for integrated intelligent agents (J. Laird, 2014). In order to learn a new task, an agent must learn a variety of types of knowledge. It not only should know the description of task goals but also learn to recognize relevant objects, their features, and relationships from its sensory stream. It must learn to manipulate its environment to make progress toward its goals. It also needs to acquire task parameters and specifications. Prior work (Chernova and Thomaz, 2014) on learning from demonstration, dialog, and reinforcement has addressed the ITL problem in parts. Several initiatives have focused on acquisition of control policies from either human generated embodied traces or reward. However, very few have studied learning comprehensive representations of tasks from scratch.

A related challenge is that of supporting natural interactions with human collaborators. Most commercially available agents and robots rely on menu-driven interactions that are completely controlled by the human user. To design autonomous agents that can meaningfully interact with humans to collaborate on tasks, more natural interaction modalities such as language, gestures, sketching, or demonstrations have to be explored. Both *Siri* and *Google Now* have taken encouraging steps in this direction by relying on spoken language. However, their interaction paradigm is unidirectional, and they merely respond to human initiated queries. Furthermore, they are constrained to pre-programmed behaviors. An agent that learns effectively and efficiently by interacting with humans must assume control of interactions on occasion and guide the conversation for its own learning.

Mixed-initiative, task-oriented dialog arises naturally in scenarios where an expert

guides a novice to execute a novel task. This dialog is rich in useful information identifying task relevant features, decomposition structure, goals, and constituent actions. In comparison with other interaction modalities such as embodied traces or rewards that can only encode very specific information (execution trajectory or value judgment), the linguistic modality is extremely expressive. It can be used to communicate a variety of information, and therefore, is useful in developing strong and flexible task learning paradigms. However, it also poses a significant challenge. As the communication is *high-level*, the agent must translate the linguistic symbols to representations that it uses for perceiving, reasoning about, and manipulating its world. The contextual flexibility of human language and the ambiguities pervasive in it make this a hard problem.

This thesis addresses the problem of interactive task learning from human-agent instructional dialog. We introduce a learning paradigm based on linguistic communication called *situated interactive instruction* (SII). In the SII approach, a human instructor and an agent learner are simultaneously embedded in a shared environment. The shared perceptions and common sense knowledge about the world provides a *common ground* through which the dialog is situated in aspects of the world. The human instructor gives the agent instructions to execute new tasks. The agent grounds linguistic symbols in the instructions to objects, their perceptual attributes, spatial relationships, and tasks definitions. Through this *grounding* process, the agent extracts specific examples from the environment that form the basis of learning. The agent effectively combines knowledge in interactions with its experience in the environment to extract generally applicable knowledge and adds it to its repertoire. The task structure, parameters, and execution knowledge so acquired crucially provide the means through which the semantics of verbs are grounded in the physical properties of the world. Grounding verbs in task representation acquired from the instructor contributes to expanding the common ground between the agent and the instructor. This not only allows the agent to communicate about tasks but also aids in learning complex hierarchical tasks. The interactions are mixed-initiative and distribute

the onus of learning between both participants.

More specifically, this thesis proposes answers to the following questions.

- *How can verbs be grounded in task goals and execution knowledge?* Answering this question is critical for communicating about tasks and producing behavior in response to task commands (*imperatives*) such as *set the table*. The challenge key to this question is identifying a representation which encodes lexical, semantic, and procedural aspects of verbs and tasks that can be realized in a computational system.

- *How can task commands be understood?* This questions deals with grounded interpretation of language - generating meaning by connecting words to non-linguistic knowledge of the world. Linguistic interactions are contextual, flexible, and ambiguous which makes understanding even simple imperatives a significant challenge.

- *How can task-oriented linguistic interaction be sustained?* In order to learn task representations, the agent must acquire various aspects such as their parameters, goals, execution policy, etc. This may involve extended conversations about these aspects. In order to learn useful knowledge from interactions, the agent should be able to sustain and direct these conversations.

- *How can task goals, structure, and execution knowledge be learned interactively?* Learning the structure of novel tasks and how to execute them is a challenging computational problem that requires acquiring a variety of knowledge including goal definitions and hierarchical control information. Additionally the data available to learn from is sparse, necessitating knowledge-intensive learning paradigms.

This thesis can also be characterized as a case study in designing complex agents that have several intelligent capabilities. The methods in this thesis developed in answering the questions above are integrated in Rosie, a generally instructable agent. Rosie can

interactively learn a diverse variety of concepts including perceptual categorization and classification (Mohan, Mininger, Kirk, et al., 2012), spatial composition (Mohan, Mininger, Kirk, et al., 2012), hierarchical tasks (Mohan and J. Laird, 2014), and simple games (Kirk and J. Laird, 2014).

1.1 Research approach

We approach the challenge of designing interactive learning agents with the following three strategies.

1.1.1 An Integrated Account

Developing a generally intelligent artificial agent has proved to be an incredibly hard challenge. A strategy that has been successful is to divide the problem of general AI into smaller problem areas. This has led to tremendous successes in various sub-fields of AI from recommender systems to autonomous robot navigation. Recent advances have led to methods in several domains that can handle real-world complexity. However, there has been limited work in studying how algorithms and methods developed in these sub-fields can be integrated into a consistent framework for end-to-end intelligent behavior.

The problem of learning new tasks from human-agent dialog necessitates the integration of several intelligent capabilities including language comprehension and generation, interaction and dialog, perception and actuation, and learning. Each of these is a significant research challenge in its own regard. In order to make this cross-capability integration tractable, we focus on task learning in a simple robotic domain (described in detail in Chapter 4). The domain consists of a robotic arm that can manipulate blocks on its tabletop workspace. This strategy simplifies the perceptual and actuation challenges, allowing us to study representations and processes useful for interactive task learning while maintaining end-to-end behavior. Our research is embedded in a larger research initiative that

investigates methods to learn a variety of concepts ground-up in a single integrated agent. The concepts that can be learned include perceptual attributes and spatial relations, tasks, and simple games.

In studying situated language comprehension, we restrict the grammar and vocabulary of communication to *concrete* nouns, adjectives, and verbs that can be grounded into perceptual data and control hierarchies. This allows us to explore connections between language and other aspects of intelligent behavior.

1.1.2 Architectural Implementation

Designing an agent with several intelligent capabilities is challenging and necessitates an agent architecture that provides reliable and efficient mechanisms for perceptions, actuation, memory, decision making and learning. The representations and methods described in this paper are implemented as components of Rosie (Mohan, Mininger, Kirk, et al., 2012) which is developed in the Soar cognitive architecture (J. Laird, 2012). Soar incorporates various learning, memory, and control mechanisms and is committed to reactive behavior (50 ms perceive-decide-act cycle), online learning, and diverse modality-specific representations. This makes Soar a suitable AI architecture for use on robots and in interactive learning. An analysis of memories and processing useful in designing Rosie is presented in Chapter 4.

1.1.3 Qualitative and Functional Evaluation

Our methods are motivated by and evaluated on desirable characteristics of interactive learning agents. Chapter 2 identifies the desiderata for an intelligent agent that can maintain a situated task-oriented dialog and learn from this experience. This is based on prior work in various fields that have studied human-human task-oriented dialog. The methods introduced in the later chapter are analyzed on how close they are to desirable behavior

of interactive learners.

For functional evaluations, the primary metric used is the number of human-agent utterances required to learn or execute a task. A competent interactive learner bears a significant responsibility for its own learning and takes initiative in generalizing its experience, applying its background knowledge to learning, and exploring its environment. Consequently, it requires less interaction with its instructor in comparison with a passive learner that relies on the instructor to structure instructions and provide examples for appropriate generalization.

This thesis takes an agent-oriented view of learning from human-agent dialog, addressing integration of various intelligent capabilities that will allow an agent to sustain a task-oriented dialog and learn useful representations from it. Therefore, the evaluations are functional, focusing on correctness and efficiency of task learning. In future, we will evaluate our methods in human-robot/agent interaction contexts to explore variability in human instruction and to develop methods that are robust to these variations.

1.2 Looking Ahead

Chapter 2 introduces *situated interactive instruction* (SII) - our approach to learning new tasks. We begin by studying the properties of task-oriented dialog which forms the basis of SII. Based on these properties, we then derive design requirements for agents that can learn from SII.

Chapter 3 reviews literature related to our research and positions this thesis with respect to contributions made by prior work.

In Chapter 4 we present an overview of Rosie, an SII agent. We begin with a brief primer to the underlying cognitive architecture, Soar, and analyze the degree to which the architecture satisfies SII requirements. Next, we describe our experimental robotic domain and how it is interfaced with *Rosie*.

Chapter 5 presents a semantic and task-oriented analysis of verbs used to describe common domestic tasks. This analysis is then used to motivate a *mixed-modality* representation for verbs that encodes lexical, semantic, and task-oriented knowledge.

Chapter 6 proposes a computational model of situated language comprehension based on the Indexical Hypothesis Glenberg and Robertson, 1999. The Indexical Model generates meaning representations by translating amodal linguistic symbols to modal representations of beliefs, knowledge, and experience external to the linguistic system. It incorporates multiple information sources including perceptions, domain knowledge, and short-term and long-term experiences during comprehension to alleviate various semantic ambiguities.

Chapter 7 describes a computational model for maintaining task-oriented flexible human-agent dialog. We show that the interaction model is sensitive to the agent's knowledge state and accommodates instructor-driven and learner-driven learning strategy.

In Chapter 8, we study learning goal-oriented hierarchical tasks from SII. We frame acquisition of novel tasks as an explanation-based learning (EBL) problem and propose an interactive learning variant of EBL. We show that our approach can exploit information in situated instructions along with the domain knowledge to demonstrate fast generalization on several tasks. The knowledge acquired transfers across structurally similar tasks. Finally, we show that our approach seamlessly combines agent-driven exploration with instructions for mixed-initiative learning.

Chapter 9 re-iterates the contributions of this thesis and discusses open questions in interactive task learning.

Chapter 2
Situated Interactive Instruction

Task-oriented dialog arises naturally in scenarios when people collaborate on tasks. It is also common in instructional scenarios when an expert guides a novice in a new task. Research on collaborative task-oriented dialog suggests that such dialog is rich in information relevant to task execution. It may identify relevant perceptual features (Grosz and Sidner, 1986; Oviatt and Cohen, 1991; Scheutz, Cantrell, et al., 2011), subtasks (Grosz and Sidner, 1986; Bangalore et al., 2008), or goals (Grosz and Sidner, 1986; Scheutz, Cantrell, et al., 2011). It may also contain corrections (Litman and Allen, 1987), clarification (Litman and Allen, 1990) or preconditions for task execution (Lochbaum, 1998). Task dialog, therefore, can serve as an important tool to learn a new task and forms the basis of our *situated interactive instruction* (SII) approach to interactive learning.

2.1 Situated Interactive Instruction

In SII, a *instructor* or *mentor* and a learner are simultaneously embedded in a shared environment and the instructor guides the novice to execute a novel task. The instructor and the learner form a system of joint learning, which distributes the onus of learning between both participants. The instructor takes initiative in identifying relevant perceptual features, objects, and relationships in the shared environment and in structuring and decomposing the task. The learner takes initiative in actively comprehending and applying the instructions to the current situation, in applying common-sense reasoning and

experience to understand ambiguous instructions, in exploring its environment, and in posing relevant queries that elicit instructions useful in making progress in the task.

To design an agent that can learn from SII, several complex aspects of intelligence and cognition have to be addressed. Not only must the agent maintain an ongoing, mixed-initiative dialog with the instructor, it must also act in the world in accordance with the instructions, and learn from its experience of the world. The challenge of developing SII learners requires studying how language comprehension, dialog management, perception, decision-making, action, and experience-driven learning can be integrated in a single agent architecture. In order to guide the design of our agent, its capabilities, and their integration, we first study the properties of general situated interactive instruction in Section 2.2 and then, derive the design requirements in 2.3 along with identifying the specific aspects this thesis addresses.

2.2 Properties

The properties of an interactive learner can be characterized along three dimensions. The first dimension, *modality*, pertains to how the information in interactions is encoded. The second, *control*, pertains to which participant of human-agent interaction has the onus of learning. The third dimension *data*, characterizes data available to the learner to learn from. Below we characterize SII along these dimensions.

2.2.1 Modality

Information in human-agent interaction can be encoded in various ways. In typical learning from demonstration approaches (Argall et al., 2009; Chernova and Thomaz, 2014), the agent is given embodied traces of desired behavior either through teleoperation or by direct manipulation of its actuators. The traces are composed of state-action pairs from which the agent can induce a control policy. Interactive reinforcement learning methods

(Maclin and Shavlik, 1996; Thomaz et al., 2006; Knox and Stone, 2010; Griffith et al., 2013) have looked at interaction as a medium for rewarding the agent for behavior. Both classes of interactions are *low-level* and are very close to agent's reasoning units (state, action, rewards). Other approaches (Rybski et al., 2007; Cantrell, Talamadupula, et al., 2012; Meriçli et al., 2014) have looked at *high-level* dialog-type interactions. Interactions in SII are high-level and rely on:

P1 **Language**: Language in task-oriented dialog is referential and is used to identify and bring to the attention of the collaborator the objects of interests, actions to be taken in the environment, useful relationships between objects, and feedback from the environment. For example,

> *"so you see that thing on the wall on the right"* (Byron and Fosler-Lussier, 2005)
> *"You should be seeing a door in front of you"* (Scheutz, Cantrell, et al., 2011)

Through such references, the speaker and the hearer accumulate a *common ground* of shared beliefs and mental representations. This allows them to *situate* the communication in the current task and environmental state. Comprehending instructions requires the agent to translate the amodal symbols in the linguistic utterance to its modal beliefs about perceptual state, domain knowledge, and experiences. Even very simple instructions can be linguistically complex and ambiguous requiring the hearer to use non-linguistic context and domain knowledge for unambiguous interpretations. Disfluencies, continuations, and ungrammatical constructions are common (Scheutz, Cantrell, et al., 2011) .

P2 **Multi-modal**: Although language is the primary modality in task-oriented dialog, other modalities (Cassell et al., 1999) may be employed to convey information critical for task performance. Gestures are important for establishing object references and may be used to communicate size or distance. Eye-gaze and gestures are useful in conveying understanding, confusion, or distress.

2.2.2 Control

Human-agent interaction for learning can be viewed on a continuum of instructor/agent control. Learning from demonstration or by imitation focuses on instructor-controlled interactions, in which the instructor provides examples of task performance, object recognition and categorization, etc. The agent observes these examples, maps the performance onto its own capabilities and induces knowledge, goals, or reward functions. Such systems place the onus of learning completely on the instructor, requiring the instructor to model the learner and provide good samples from the feature space so that the learner can acquire general hypotheses. Other approaches (Maclin and Shavlik, 1996; Thomaz et al., 2006; Knox and Stone, 2010; Knox and Stone, 2012; Griffith et al., 2013) incorporate human feedback as a reward in a reinforcement learning architecture, letting the agent (or the agent designer) control learning. A few others (Allen et al., 2007; Huffman, 1994) have explored distributing initiative between the instructor and the agent. SII can be characterized as:

P3 **Mixed-initiative**: Task-oriented dialog is mixed-initiative, flexible, and collaborative. Participants advance the dialog in accordance with their intentions and goals and comply with each other's requests. Dialog provides an opportunity for mixed-control of learning. An example from Grosz and Sidner (1986) (annotation ours):

> E: *First you have to remove the flywheel.*
> A: *How do I remove the flywheel?* (learner initiative)
> E: *First, loosen the two allen head setscrews holding it to the shaft, then pull it off.*
> A: *OK. I can only find one screw. Where's the other one?* (learner initiative)
> ...
> E: *Use the wheelpuller. Do you know how to use it?* (instructor initiative)
> A: *No.*
> E: *Do you know what it looks like?* (instructor initiative)
> A: *Yes.*
> E: *Show it to me please.* (instructor initiative)
> A: *OK.*
> E: *Good, Loosen the screw in the center and place the jaws around the hub*

of the wheel, then tighten the screw onto the center of the shaft. The wheel should slide off.

Mixed initiative in task dialog offers some advantages. A learner that can assume control on occasion can guide its own learning by requesting clarifications, asking for missing information, exploring its environment, and correcting instructor's response based on its own understanding of the state. A learner that can relinquish control on occasion, can take advantage of instructor's knowledge of task structuring and goal decomposition. In a mixed control setting, the instructor can rely on the learner to provide information about its state and the environment. The instructor can verify the learner's learning by questioning the learner and provide corrections without having to continuously model the state of learner's knowledge.

2.2.3 Content

Information content in interactive learning can vary along several dimensions including what information is encoded, how its presented, and how much of it is available to learn from. It can be characterized as:

P4 **Diverse information types**: As noted earlier, task-oriented dialog contains a variety of information including task goals, decomposition substructure, relevant perceptual features etc. The agent must continuously reason about how the current instruction relates to the task in order to use it effectively.

P5 **Incremental**: Task instructions rarely contain the complete procedure to do task in a single interaction. Rather, the dialog unfolds as the listener acts in the environment in response to the instructions, resulting in complex interactions. For example, an instructional dialog for following a particular path in the environment is given below (Scheutz, Cantrell, et al., 2011).

Instructor: *OK, continue to walk straight.*
Robot (continuing straight): *OK.*
Instructor: *You should be seeing a door in front of you.*
Robot (looking out for a door): *Yes.*
Instructor: *Good, go through that door.*
Robot (moving through the door): *OK. I'm through the door.*
Instructor: *Alright. Keep going. There should be a whiteboard.*
Robot (looking for whiteboard): *OK, I'm not seeing it yet. There it is.*
Instructor: *Great, then you should see an intersection, go there.*
Robot (looking out for an intersection while moving): *Got it, OK.*

Such interaction requires that sensing, language understanding, and behavior must be intertwined and must be performed online. The information provided for performing the task is spread out in time. To learn from the interaction, the agent must maintain the memory of task performance that is available for later analysis and generalization.

P6 **Situation Specific**: Task-oriented dialog is specific to the current situation observable to the speaker and the hearer.

> *"Good, go through that door."* (Scheutz, Cantrell, et al., 2011)
> *"And put it so that it's covering the hole in the bottom of that little cap."* (Oviatt and Cohen, 1991).

Therefore, the information extracted from the dialog applies to only a few scenarios and does not encode how tasks can be executed in general.

P7 **Sparse**: In general, learning requires a large amount of data. A reinforcement learning architecture may need several iterations of taking an action and observing the reward to induce a good policy. A supervised learning architecture may need several labeled examples for inducing general concept definitions. However, human time is costly and numerous, repetitive interactions about the task are undesirable. Consequently, large amount of data may not be readily available while learning

from human-agent interactions. Therefore, the agent must implement methods that learn from a few instructions and examples.

2.3 Desiderata

Based on the properties of instructional dialog identified above, we now derive the desiderata for SII learners. We refer to these desiderata throughout the remainder of this dissertation while studying the related work and developing the functional characteristics of computational models well as evaluating our work.

D1 *Real-time reactivity*. The agent is embodied as a robot and must maintain interactivity with its dynamic environment along with supporting a real-time[1] dialog with a human collaborator. This requires that the agent be reactive to changes in its sensory input and respond in real-time.

2.3.1 Situated Comprehension

As the human-agent interaction in SII is high-level (P1) and multi-modal (P2), the agent must implement a comprehension model that extracts useful information from instructor's utterances. The comprehension model must be:

D2 **Referential**. It must implement a theory of translating amodal linguistic symbols used for communication to modal representations of beliefs, knowledge, and experience that are external to the linguistic system.

D3 **Integrative**. Human language is highly contextual and relies on several non-linguistic sources to convey meaning. To successfully comprehend language, a model must exploit multiple information sources, including perceptions, domain knowledge,

[1]Real-time in this context means that the response to instructions and questions in generated in less than 500 milliseconds so that the human instructor is not disengaged.

common-sense knowledge, and short- and long-term experiences. It should also readily incorporate information from non-verbal communication such as gestures and eye gazes.

D4 **Active**. All reasoning and knowledge access must be performed online as the interaction progresses. This processing should inform further communication with the collaborator and learning.

D5 **Expandable**. As the agent gathers knowledge and experience of its environment, it should use be able to use this to comprehend instructor's utterances.

D6 **Incremental**. The model must build up the meaning representation as each word is processed. Incremental processing generates expectations about likely continuations and informs linguistic, speech perception and provides robustness to noise.

2.3.2 Integrative Interaction

The agent must maintain a continual, online interaction with its human instructor. To facilitate this, an interaction model is required that is:

D7 **Task-oriented**. The interaction model should capture the structure of task oriented communication, provide discourse context for resolving ambiguities, correct interpretation of instructions (P4), and organize dialog so that it is useful in task execution and learning. It must be useful for generating learning-oriented interpretations of instructions and for asking questions relevant to the task execution.

D8 **Integrative**. The model should reason about a joint space of comprehension, interaction, behavior, and learning in order to interpret and advance the ongoing dialog.

D9 **Flexible**. SII affords mixed-control of learning (P3). To take advantage of this, the interaction model must be flexible. The model should allow both participants of SII to change the focus of communication (*flexible initiation*) regarding various aspects

(perceptual, spatial, semantic, procedural) of task (*flexible content*). Furthermore, the model should not impose any requirements on the order in which task-relevant knowledge is presented (*flexible ordering*) but rely on what is required for making progress in the current task.

2.3.3 Incremental Learning

The characterization of data in task-oriented dialog requires that the learning paradigm be -

D10 **Multi-method**. It must employ methods that can learn useful knowledge from different types of information in SII (P4).

D11 **Assimilative**. The paradigm must collect information that is presented incrementally (P5) in the dialog and induce an integrated task representation. The acquired knowledge should integrate with prior knowledge that either has been pre-encoded or has been acquired through other experience with the environment.

D12 **Multi-task**. The learning paradigm should be useful in learning a variety of tasks.

D13 **General**. As the data is situation-specific (P6), the paradigm generalize specific instruction to unseen scenarios that may arise during task performance in future.

D14 **Fast**. Given that the data available for learning is sparse (P7), knowledge-rich learning algorithms must be employed. The paradigm should be able exploit the background knowledge to learn the best generalization from specific examples.

D15 **Transferable**. The sparsity of data (P7) also motivates exploiting the similarity between tasks to transfer knowledge from a known task to a new one.

D16 **Active**. As task-dialog affords mixed control, the agent must be an active participant in its own learning. The agent must detect when it lacks knowledge to make

progress on the task and take actions to acquire that knowledge. This might include exploring its environment, model, or asking a relevant question to the instructor.

D17 **Online**. The paradigm must acquire knowledge online, as the dialog progresses, without interrupting other processes.

Chapter 3

Related Work

Various fields in AI have studied the computational challenges relevant to the design of SII learners. Therefore, our work can be compared to a wide variety of approaches along different dimensions. To present a coherent analysis of the related work, we use the desiderata for SII learners identified earlier. The following sections briefly summarize and analyze the previous work categorized by its primary motivation. Section 3.1 presents work that focuses on grounding language in extra-linguistic representations, section 3.2 describes the progress made in sustaining human-agent dialog, and section 3.3 summarizes the prior work in interactive learning. This thesis is uniquely placed as it presents an integrated agent design addressing each of these categories.

3.1 Grounded Language Semantics

Designing a linguistic faculty for intelligent agents has been one of the original goals of AI. The research pursued to approach this goal can be organized under the following broad categories. The prior work reviewed in this section addresses the *situated comprehension* desiderata (D2-D6).

3.1.1 Computational Linguistics

Research on semantics in computational linguistics and natural language processing can be broadly categorized into three distinct groups, *formal*, *distributional*, and *grounded* se-

mantics. While the earlier two approaches have been well studied in the literature, the last approach has recently gained momentum. The formal approaches to semantics represent meaning as amodal first-order logic symbols and statements. Although this lets an agent incorporate extra-linguistic knowledge during comprehension through inference, the symbols and predicates are not grounded in the environment. Distributional semantics only incorporates linguistic contexts with no explicit groundings to the observations from the environment.

Work on grounded language acquisition has taken an application-oriented approach and has developed solutions for applications including navigational tasks (MacMahon et al., 2006; D. Chen and Mooney, 2011) and RoboCup sportscasting (Liang et al., 2009). These projects have focused on acquisition (D5) of grounded lexicon and semantic parsers from aligned corpora of agent behavior and the text that describes it. There are several reasons for why such approaches cannot be used to design collaborative agents that engage in situated communication. These methods apply data-intensive learning paradigms offline. Although this provides guarantees about robustness to noise in linguistic input, the methods cannot be adapted to learn online (D17). Failure in comprehension is reported but is not used to drive communication or learning (D4). Further, the work proposes that the entire complexity of language comprehension be encoded in a semantic parser and does not address the use of reasoning mechanisms and background conceptual knowledge for the purposes of language comprehension. Finally, these approaches assume a propositional state and action representations. This simple representation of the world state and dynamics poses problems in adapting the comprehension model to agents embedded in physical environments that require complex, relational representations for reasoning and action. Additionally, these approaches do not provide insights into the role of non-linguistic context on language processing (D3).

3.1.2 Integrated AI Systems

There has been a long history of studying language understanding and generation in AI systems going back to SHRDLU (Winograd, 1972). SHRDLU was an early attempt to design an intelligent agent that could understand and generate natural language referring to objects and actions (D2) in a simple virtual blocks world. It performed semantic interpretation by attaching short procedures to lexical units. It demonstrated simple learning as the user could define compositions of blocks (such as a tower) that the system would remember and could construct and answer questions about (minimally addresses D5). However, the system did not learn new procedures or perceptual knowledge, constraining the system to pre-programmed behaviors and features. The system's reliance on formal logic for internal representation not only made it harder to extend its capabilities but also made it brittle and unsuitable for robotic domains. Nevertheless, SHRDLU pioneered the view that language processing for agents greatly benefits from general problem solving capabilities, which this thesis also propounds.

Ongoing work on Direct Memory Access Parsing (DMAP; Livingston and Riesbeck, 2009) studies the utility of incorporating information from ontological and instance-based inference for linguistic processing in the context of learning by reading. DMAP incrementally integrates conceptual memory (available through an ontology) during parsing, which can reduce the number of ambiguous interpretations and reference resolution. DMAP has several desirable properties. It is referential (D2), integrative (D3), and active (D4) but it has not been investigated in human-agent interaction contexts.

Other work has addressed the challenge of situated language processing for human-agent interaction. Scheutz, Eberhard, et al. (2004) presented a visually-grounded, filter-based model for reference resolution that is implemented on a robot with audio and video inputs. Ambiguities are resolved by accounting for attentional context arising from fixations in the work area. In a related work, Kruijff et al. (2007) demonstrated incremental parsing at multiple levels that includes non-linguistic contexts, such as the ongoing dialog

and declarative pre-encoded selectional restrictions along with visual semantics. Apart from being referential and integrative, these projects address issues that arise in spoken dialog processing and online, incremental comprehension (D6). Brenner et al. (2007) describe how action commands can be interpreted in a task-oriented fashion to identify and instantiate goals and plans. This model brings in knowledge about initial state and goal descriptions that are relevant to generating and executing a plan.

3.1.3 Robotics

In robotics, language comprehension has been studied within the context of grounding verbs in reasoning for force-dynamic properties (Siskind, 2001), describing a visual scene (Roy, 2002), learning and describing perceptual features (Matuszek et al., 2012), understanding descriptions of a scene (Gorniak and Roy, 2004), understanding spatial directions (Kollar et al., 2010), and understanding natural language commands for navigation (Tellex et al., 2011). These comprehension models work with the complex state and action representations required for reasoning about physical worlds (D2). Their primary focus has been on the acquisition of grounding models through offline learning from human-generated descriptions of robot's perceptions or behavior. The agents are prone to failure if their training is insufficient for grounding a novel instruction. An interactive agent on the other hand will switch to learning mode if it is unable to comprehend the instruction. It is unclear if such data intensive, corpus-based learning paradigms can be effectively incorporated in online and incremental human-agent interactions.

3.2 Human-Agent Dialog

Conversational systems have been investigated both for virtual agents/characters and for robots operating in physical world. While the research on conversational virtual agents has focused on the desiderata for *integrative interaction* (D7 - D9), research on conversa-

tional robots has also addressed some desiderata for *situated comprehension* (D2 - D6).

3.2.1 Virtual Agents

Research on virtual conversational agents (Cassell, 2000a) has progressed in two separate themes. A body of research (Cassell, 2000b; Poggi et al., 2005; Swartout et al., 2006) has focused on *believable* agents, usually embodied in human avatars. The agents developed have several similarities to human conversational behavior including recognizing and generating verbal and non-verbal interactions; dealing with conversational functions such as turn-taking, feedback, and repair mechanisms; giving signals that indicate the state of conversation; and, expressing emotions. Typically, the dialog strategies developed are not task-oriented (D7).

A separate body of research has looked at conversational agents engaged in collaborative tasks with humans. Although, these agents tend to less *believable*, they are more relevant to this thesis as they address desiderata D7. Rich and Sidner (1998) demonstrated an application-independent collaboration manager that allows an agent to provide intelligent, mixed-initiative assistance for an air-travel application. The agent was a planning system that interacted with a human user to determine the constraints and goals of their air-travel and managed reservations. Task-oriented virtual agents have been recently incorporated in smartphone architectures. Siri (Apple, 2013) and Google Now (Google, 2013) collaborate with users to search the web, user's emails, calendars etc. and retrieve relevant information.

3.2.2 Conversational Robots

Recent advancements in AI, vision, and robotics has made real-time interactions with robots feasible. Consequently, several research projects have begun to look at linguistic human-robot interactions. Cantrell, Scheutz, et al. (2010) demonstrate a natural language

understanding architecture for human-robot interaction that integrates speech recognition, incremental parsing, incremental semantic analysis and situated reference resolution. The semantic interpretation of sentences is based on lambda representations and combinatory categorial grammar.

Other research has looked at the consequences of embodiment in physical worlds in human-robot dialog and the various challenges it presents. Human and robots have significant differences in perceiving the shared environment and their representations may be misaligned. Chai et al. (2014) investigate how a robot operating with impoverished representations can use its communicative experience to mediate a shared perceptual basis. Deits et al. (2013) demonstrate that dialog strategies implemented using an information-theoretic framework are useful in reducing uncertainty in language understanding. Both of these projects demonstrate referential (D2) and active (D4) comprehension along with implementing flexible (D9) dialog strategies. Other work such as Mutlu et al. (2009) has studied the role of gaze cues in human-robot dialog (D3).

3.3 Interactive Learning

Research on interactive learning has explored learning from various types of interaction modalities. *Low-level interactions* include embodied traces provided through teleoperation or kinesthetic training, observed traces from similarly embodied agents, and rewards. *High-level interactions* typically involve the use of symbolic language or gestures to convey information. This body of work primarily addresses the *incremental learning* desiderata (D10-D17).

3.3.1 Learning from Low-level Interactions

Learning from embodied demonstration traces has historically focused on learning control policies or *skills* (Sammut et al., 1992; Atkeson and Schaal, 1997). The demonstration traces

are obtained either through teleoperation or shadowing. These traces consists of relevant state-action pairs from which the agent can derive control policies. Argall et al. (2009) and Chernova and Thomaz (2014) provide a more comprehensive survey of skill learning from demonstration.

A few prior approaches have studied learning tasks from demonstration. They frame task learning as learning compositions of known primitive actions. Bentivegna et al. (2004) proposed a framework that learns subgoals from segmented observed data. The system can use its experience to optimize its policy of selecting subgoals. Grollman and Jenkins (2010) formulate the problem of task acquisition as inferring a finite-state machine from segmented observations of a demonstrator performing the task. While the former is a batch learning system, the latter focuses on learning interactively during performance. Although powerful, these methods are instructor-driven (D9) and rely on the instructor to provide good samples for appropriate generality. Further, these methods usually require several demonstrations (D14) and do not make any claims about generality (D13) and transferability (D15).

Other work (Maclin and Shavlik, 1996; Thomaz et al., 2006; Knox and Stone, 2010; Griffith et al., 2013) has explored instructions as rewards in a reinforcement learning framework. Usually, the agent is pre-encoded with state descriptors and actions. During its exploration of its environments an intelligent observer can reward the agent for perceived good behavior. This reward is used to optimize an action selection policy. These methods have focused on only a single aspect of task learning and cannot be extended to learning goals or action pre-conditions, models etc. (D10).

3.3.2 Learning from High-level Interactions

Prior work on task learning from spoken dialog (Rybski et al., 2007; Cantrell, Talamadupula, et al., 2012; Meriçli et al., 2014) addresses learning task procedures. Through linguistic constructions (obtained by imposing additional syntactic constraints on natural language),

these agents can be programmed to execute procedures defined over their primitive, pre-encoded actions. An example from Rybski et al., 2007 is below.

```
>> When I say deliver message
>> If Person1 is present
>> Give message to Person1
>> Otherwise
>> If Person2 is present
>> Give message to Person2
>> Otherwise
>> Report message delivery failure
>> Goto home
```

Using domain-general mechanisms, this instruction is translated into procedures that are added to the agent's repertoire. In these approaches too, the onus of learning is on the instructor (D9, P3) who must explicitly identify the pre-conditions, termination criterion, and the procedure of doing a task. Furthermore, the instructions deviate from what constitutes a natural task-oriented dialog which usually involves executing a specific task (P6). Instead, explicit programming control structures such as the *if-then* block (in the example above) are included in instructions.

Others have looked at learning from task-oriented interactions. Early work done on learning from instruction by Huffman and J. Laird (1995) demonstrated how instructions can be useful in learning different types of knowledge in the problem space computational model (PSCM) for a virtual agent. Our work can be characterized as a significant extension to this work. We present an integrated account of task learning, comprehension, and interaction management. The prior work neither identified the space in which tasks vary nor identified the representations sufficient for representing a variety of tasks (D12), which is central to this thesis. Further, the prior work was silent on the generality and transferability of acquired knowledge.

Nicolescu and Mataric (2003) proposed learning task behavior networks from experienced demonstrations. The demonstrations can be accompanied by verbal instructions that are used to indicate moments with relevant sensory features (*here*), to induce the

learner in doing certain actions (*drop*), and signaling the beginning and end of the demonstration (*start*, *done*). The learning framework also allows *practice* trials in which the instructor can provide feedback and corrections to the robot's performance using speech. The approach integrates language with demonstration (P2, D10) which results in quicker generalization of task representations. The authors do not comment on if their task representation can be used for encoding a variety of tasks (D12) or if acquired networks can be shared between multiple tasks (D15).

Allen et al. (2007) demonstrated a virtual learning agent that learns executable task models from a single collaborative session of demonstration, learning, and dialog. The human teacher provides a set of tutorial instructions accompanied with related demonstrations in a shared environment, from which the agent acquires task models. The initiative of learning is on the human user. However, the agent controls certain aspects of its learning by making generalizations about certain tasks without requiring the human to provide numerous examples (D15). This approach is novel in that it combines learning from demonstrations with explicit instructions (P2, D10). Although the approach described is powerful, the authors do not make any claims about generality of their methods and if their methods are sufficient for learning a variety of tasks. The agent lived in a virtual environment which simplified the problem of grounding words to elements of the world.

X. Chen et al. (2010) describe a unified agent architecture for human-robot collaboration that combines natural language processing and common sense reasoning. They developed a planning agent that relies on communication with the human to acquire further information about under-specified tasks. The agent also demonstrates limited learning by acquiring novel common sense rules through dialog. Although the work demonstrates integrated natural language processing, common sense reasoning, planning, and robot navigation and manipulation, it is silent on how their design could be extended to demonstrated diverse and comprehensive task learning (D10).

3.4 Summary and Discussion

In this chapter, we reviewed prior work that studies different computational challenges of SII. While these aspects - *situated comprehension*, *human-agent dialog*, *interactive learning* - have been explored individually by different AI communities, very few initiatives have looked at integrating these capabilities in a single agent which is one of the main thrusts of this thesis. Several methods proposed recently (Liang et al., 2009; Kollar et al., 2010; Tellex et al., 2011; D. Chen and Mooney, 2011) for situated comprehension rely on offline learning from a human-generated corpus. These methods are unsuitable for interactive agents that must process information and acquire knowledge online while being reactive to their world and collaborators. Similarly, work on believable conversational agents has limited relevance to designing functional learning agents. Our proposed approach for situated task-oriented dialog stresses the role of non-linguistic context for comprehension and can be categorized with work done in integrated AI systems (Scheutz, Eberhard, et al., 2004; Brenner et al., 2007; Kruijff et al., 2007) and in conversational robots (Cantrell, Scheutz, et al., 2010; Chai et al., 2014). The task learning paradigm proposed in this thesis shares goals and motivations with Huffman and J. Laird, 1995 and Allen et al., 2007. Our work makes significant extensions to these approaches by proposing and analyzing general, composable, hierarchical task representations that provide grounding to verbs.

Chapter 4

Rosie Overview

Rosie is a generally instructable agent developed in the Soar cognitive architecture (J. Laird, 2012) and embodied in a table-top robotic arm. It uses the SII approach to learn concepts such as perceptual attributes (*color*, *shape*, *size*), spatial relationships (*right-of*, *in*), tasks (*place*, *stack*), and simple games (*tower-of-hanoi*, *tic-tac-toe*). Below we describe Soar, the cognitive architecture underlying Rosie and then give a brief overview of Rosie's environment, interfaces, and processing cycle.

4.1 The Soar Cognitive Architecture

Soar is a cognitive architecture that has been applied to a wide variety of AI applications and cognitive models. Recent extensions to Soar, including episodic and semantic memories, as well as a visual-spatial system, enhance Soar's ability to support the SII approach for interactive task learning. Components relevant to Rosie's design are described in below. Later, we analyze the degree to which it satisfies the requirements of SII agents. For definitions of the terms used here refer to Appendix 1.

4.1.1 Architecture Overview

Figure 4.1 shows a structural diagram of Soar. Soar's *working memory* represents the agent's current state. Functionally, it serves as a common substrate that maintains symbolic relational representations of current and recent sensory data, current goals, and the

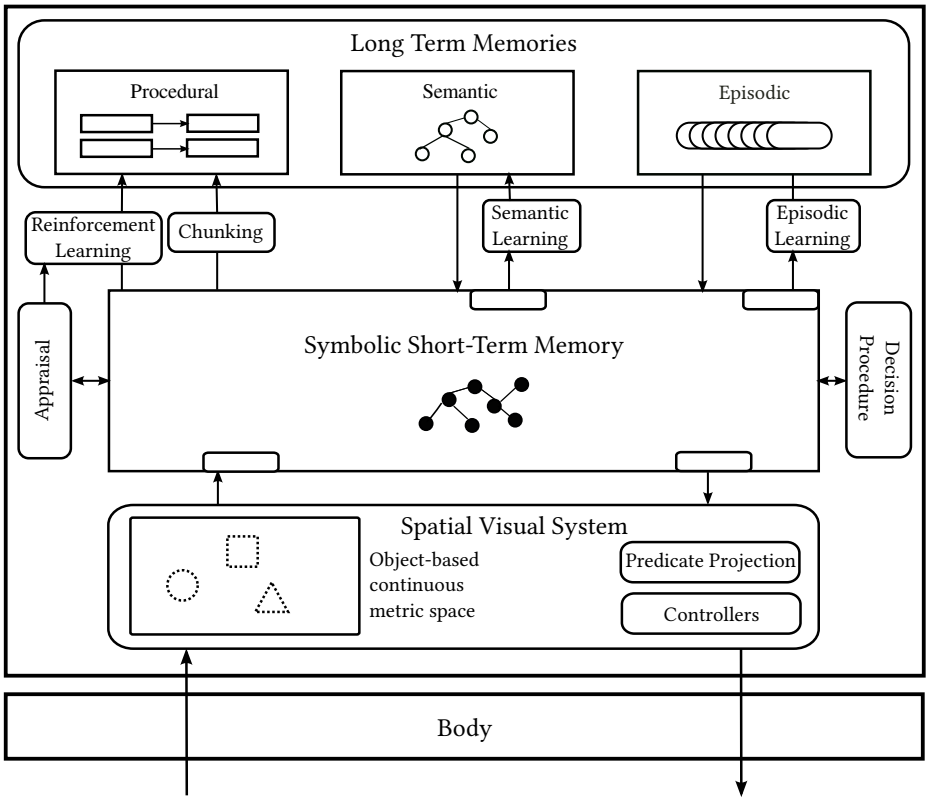

Figure 4.1: The Soar Cognitive Architecture (adapted from J. Laird, 2012)

agent's interpretation of the current situation encoded as connected, directed graph. It also provides interfaces to Soar's long-term memories and other modules. The unit of knowledge representation in working memory is a working memory element (WME).

Soar contains a task-independent *spatial visual system* (SVS) that supports translation between the continuous representations required for perception and actuation in the physical world and the symbolic, relational representations necessary for high-level reasoning. The continuous environment state is represented in SVS as a scene graph composed of discrete objects and their continuous properties. SVS computes truth-values of continuous spatial predicates about objects describing properties such as alignment, containment, etc. in response to queries issued in working memory. The set of predicates that SVS can reason about is task independent and fixed, but predicate extraction is controlled using task-specific knowledge.

Semantic memory stores context independent knowledge about the world as a directed graph. The agent can store working memory elements into semantic memory. Later, these can be retrieved by creating a cue in a working memory buffer. The best match to the cue (biased by recency and frequency) is retrieved from semantic memory to working memory. Semantic memory provides bi-directional access to knowledge. The agent can retrieve a node either by creating a cue composed of its children or by retrieving its parent.

Episodic memory stores context-dependent records of the agent's experiences. It takes snapshots of working memory (episodes) and stores them in chronological order, enabling the agent to retrieve both the context and temporal relations of past experiences. The agent can deliberately retrieve an episode by creating a cue in a working memory buffer. The best partial match (biased by recency) is retrieved and added to working memory.

Procedural memory contains the agent's knowledge of when and how to perform actions, both internal, such as accessing knowledge in long-term memories or querying SVS, and external, such as manipulating its environment. This knowledge is encoded as *if-then* rules.

At the lowest level, Soar's processing consists of matching and firing rules in the procedural memory. Unlike most rule-based systems, the locus of decision making is not the selection of a rule. Instead, Soar fires all rules in parallel. The rules propose, evaluate, or apply operators, which are the locus of decision making. Only a single operator can be selected at a time, and once an operator is selected, rules sensitive to its selection and the current context perform its actions (both internal and external) by modifying working memory. Soar's primitive decision cycle consists of the following phases: encode perceptual input, elaborate current state, propose operators, select operator, process output command, and access long-term memories/SVS.

Whenever procedural knowledge for selecting or applying an operator is incomplete or in conflict, an impasse occurs and a substate is created in which more reasoning can occur. In Soar, complex behavior arises not from complex, pre-programmed plans or se-

quential procedural knowledge, but from the interplay of the agent's knowledge (or lack thereof) and the dynamics of the environment.

Chunking is a learning mechanism, similar to explanation-based learning (Mitchell et al., 1986), that creates new rules from the reasoning that occurred in a substate. When a result is created in a substate, a rule is compiled. The conditions of this rule are the working-memory elements that existed before the substate and that were necessary for creating the result, and the actions are the result. The rule is added to procedural memory and can fire immediately. Soar also incorporates reinforcement learning that tunes operator selection strategy in accordance with intrinsic or environmental reward functions.

4.1.2 Analysis

Soar provides a variety of knowledge representations and learning mechanisms (D10). The semantic memory encodes context-independent declarative knowledge; episodic memory stores temporal changes in the agent's state; procedural memory can represent goal-driven control hierarchies; and SVS can reason about continuous of the physical world. This is useful in representing different aspects of task knowledge including goals, decompositional substructure, control hierarchies, relevant perceptual features etc along with syntax and semantics of verbs. The memories and SVS store modal representations of perceptual and spatial knowledge provide grounding to linguistic symbols (D2).

Soar adopts the *problem space computational model* (PSCM: Newell and Simon, 1972) and *bounded rationality* (Simon, 1991) as core theoretical commitments in order to maintain reactivity to environmental changes. A typical Soar decision cycle, takes much less than 50 milliseconds. Empirical evidence suggests that this is sufficient for reactive behavior in numerous domains including human-computer interaction tasks, video games, and robotics. A decision cycle may involve accessing large bodies of knowledge in long-term memories, however, each access (in the expected case) is a bounded search and is guaranteed to complete in finite time. Computationally unbounded search such as logi-

cal inference or generalized search is spread out over multiple decision cycles. This lets the agent be reactive to environmental changes and terminate unbounded computation if something immediate requires attention.

Whenever a Soar agent lacks knowledge to make progress, an impasse occurs and Soar automatically creates a substate in which the goal is to resolve the impasse. In the substate, the agent can reason about why the impasse occurred and implement a strategy to resolve it. This meta-cognitive reasoning (discussed in detail in later chapters) is useful in active situated comprehension (D4) and agent-driven learning (D16).

As noted earlier, SII presents information incrementally as the dialog unfolds. Soar's episodic memory automatically stores how the agent's state evolves temporally. The memory stores the changes in environmental state along with capturing the history of dialog. This history is available for retrospective analysis and learning a comprehensive task representation (further details are in Chapter 8).

4.2 Rosie

In the following sections, we give a brief overview of Rosie's environment and interfaces and briefly describe the interaction cycle that forms the basis of interactive learning. The details are presented in Mohan, Mininger, Kirk, et al., 2012.

4.2.1 Environment

Rosie acts in a mixed reality environment that simulates a toy kitchen. It consists of a table-top robotic arm that can manipulate small foam blocks and a Kinect camera for sensing. The workspace contains several locations that have simulated functionality. For example, a *stove* can be turned on and off, and the *pantry* can be opened and closed. This can change the state of the world. For example, when the *stove* is turned on, it changes the simulated state of an object on it to *cooked* after an appropriate delay.

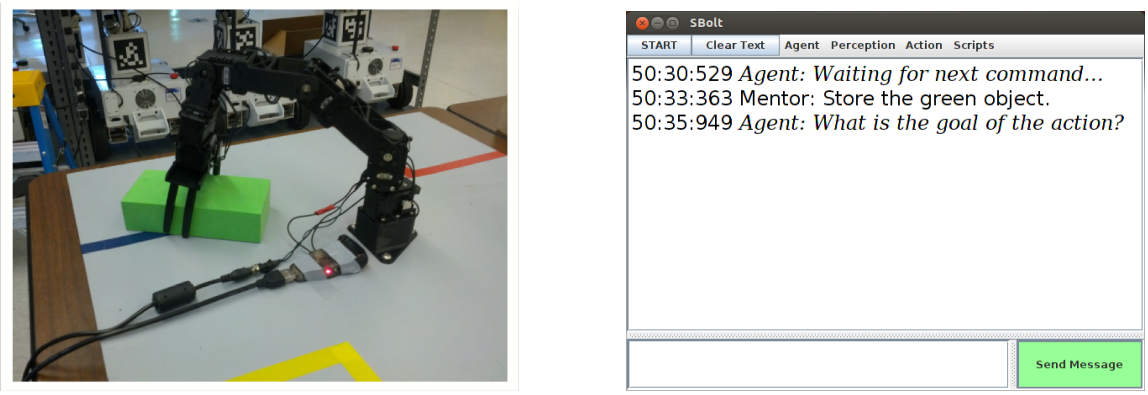

Figure 4.2: ROSIE's table-top workspace and the interaction window.

4.2.1.1 Perception

The perception system generates an object-oriented representation from the continuous data stream (3D point cloud) available from the Kinect. Each object can is associated with a vector of perceptual symbols (*red*, *large*, *triangle*) corresponding to the three perceptual properties - *color*, *size*, and *shape* - available through the camera. The perceptual symbols are extracted through K-Nearest Neighbor (*k*NN) classifiers with Gaussian weightings built for the perceptual properties. For example, a perceptual symbol `R43` may be associated with the region in the color feature space that corresponds to the word *red*. The perceptual symbols, along with position and bounding box information are provided to ROSIE. The perceptual classifiers are trained through instruction (as explained in later sections).

4.2.1.2 Actuation

To act in the world, ROSIE can send the following types of action commands to the robot controller.

- *Object manipulation* such as `pick-up (object-id)`, `place (x,y,z)` allows the agent to move objects on the work area.
- *Functional manipulation* commands such as `turn-on(stove)` change the virtual

functional state of the locations.

The low-level policies corresponding to these action commands are pre-programmed in the robot controller and the environment. The environment provides a degree of feedback indicating whether the policy was successfully executed and whether the robotic arm is free and waiting for a command. This is useful in implementing robust closed-loop control of the arm.

4.2.1.3 Interaction

Rosie can interact with the human instructor either through speech or through text in a chat interface. Speech input is converted to text using CMUSphinx (*CMUSphinx* 2014). Messages from the instructor are parsed to extract part of speech tags and syntactic structure. The messages from the agent are translated to language using templates. The instructor can also point to different objects by clicking them in a live camera feed. Supporting the entire range of complexity in human language is not a primary aim of this work, therefore, the vocabulary and grammar supported by Rosie is highly constrained. However, even in very simple constructions, some semantic ambiguity may arise due to under-constrained reference or omitting of information critical for task performance. This is studied in Chapter 6.

4.2.2 Knowledge Representation

4.2.2.1 Perceptual Knowledge

Rosie's visual knowledge is encoded in its long-term perceptual memory and semantic memory (refer to Figure 4.3). The perceptual memory accumulates training examples that are used to classify objects in terms of visual attributes: *color*, *size*, and *shape*. Each visual attribute has a k-nearest neighbor (kNN) classifier associated with it and each class within the kNN is referred to using a perceptual symbol. For example, the *color* classifier may

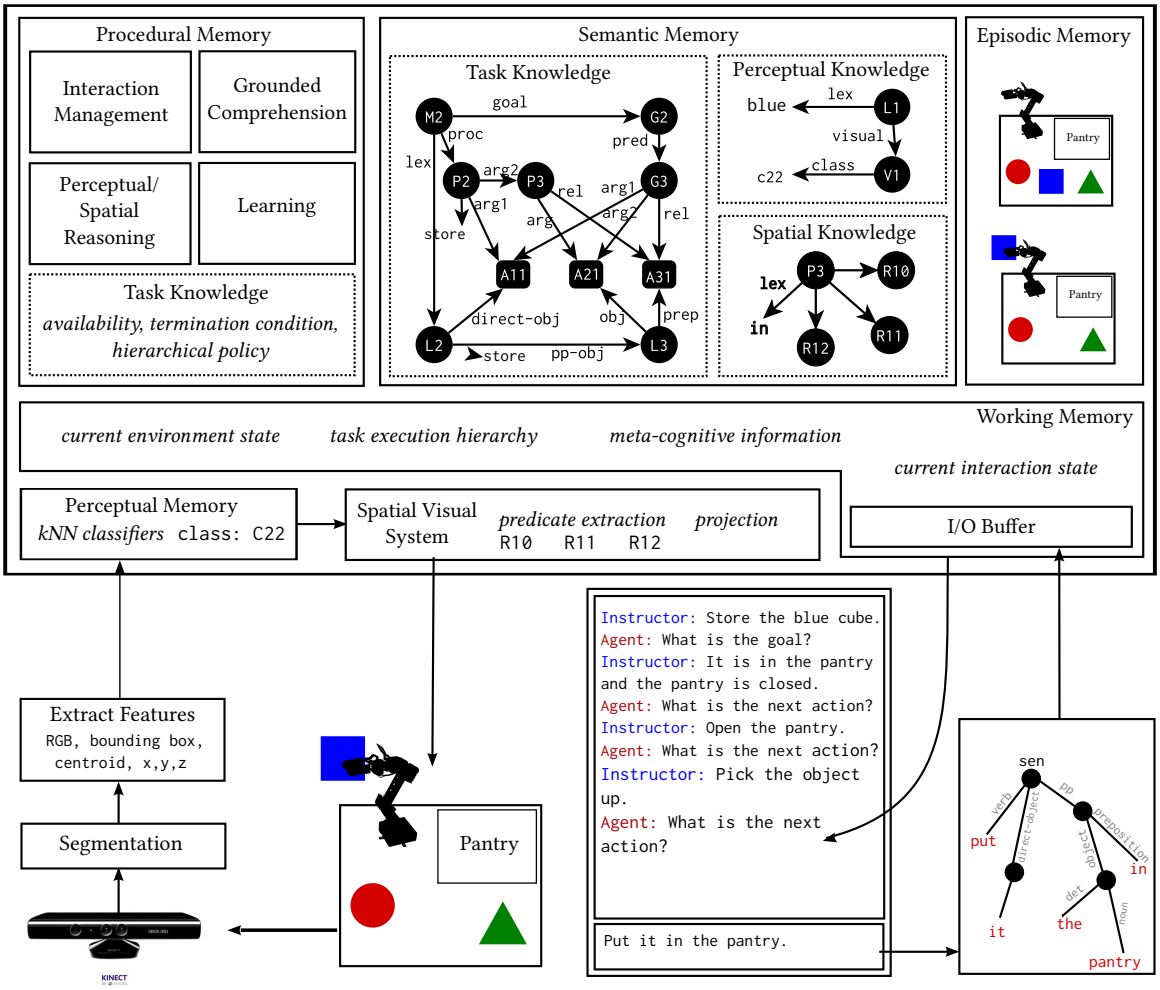

Figure 4.3: Rosie Structural Diagram

have classes labeled with perceptual symbols C22, C53, C49, each of which corresponds to a color known to Rosie. Semantic memory associates perceptual symbols with their linguistic counterparts. For example, it may maintain that the perceptual symbol C22 corresponds to the word *blue*.

All objects in the workspace are represented in the working memory and SVS. SVS maintains each object's positional information along with its bounding volume and centroid. Working memory contains symbolic information about objects. The domain is partially observable; objects occlude each other and while moving objects around, the arm may move the object out of visibility. This makes tracking objects correctly a challenge.

Perceptual reasoning implemented in Rosie's procedural memory meets this challenge to a large extent by exploiting other sources of information (Mininger, 2014).

4.2.2.2 Spatial Knowledge

Rosie's spatial knowledge is distributed between semantic memory and SVS (shown in Figure 4.3). It learns and represents spatial prepositions such as *on* and *near* as compositions of the following task-independent primitive predicates in SVS.

- The directional primitives describe how the reference and primary objects are aligned along each axis in a 3-dimensional coordinate system: `X`, `Y`, and `Z`. In relation to the reference object, the primary object can be `aligned`, `greater-than`, or `less-than`. For example, two objects on the same plane are `Z-aligned`. These relations are useful in learning prepositions that are based on spatial order, such as *right of* or *diagonal with*.

- These *distance* primitives encode the distance between the reference and the primary object along each axis. The distance is measured from the closest surface of each object. Distance-based primitives are useful in the acquisition of prepositions such as *near* or *far*.

The learned spatial relations for prepositions are represented by a logical combination of directional primitives and a distribution of distance-based primitives. The combination of directional primitives contain *conjunctions* from different axes, such as `X-less-than` *and* `Z-aligned`, and *disjunctions* on the same axis, such as `Y-aligned` *or* `Y-greater-than`. The initial teaching demonstration results in a representation with a conjunction of the current true directional primitives. Subsequent demonstrations can add disjunctive primitives. Additional demonstrations provide a distribution of distances from which a range can be calculated. Logical combinations of primitives allow the agent to acquire a wide range of complex spatial prepositions, including ones based on both distance and direction

such as *next-to*.

This representation of spatial relations is useful for reasoning about existing spatial relationships on the workspace (such as *the red cylinder is on the stove*) to generate the current state and for executing actions that establish specific spatial relationships between arguments (such as *put the red cylinder on the stove*).

4.2.2.3 State Representation

ROSIE maintains an *object-oriented* representation of the world in its working memory. An object in its view can be described as a set of perceptual features (*color*, *shape*, *size*, *volume*) and their value assignments. Formally, let $F = \{f_1, f_2, ..., f_n\}$ be the set of attributes observable in the environment. Every attribute f_i (such as *color*) has an associated domain $domain(f_i)$ (such as {`r2`, `b11`, ...}) of symbols. An object $o \in O$ can be described as the set of attribute-value pairs, $o = \{(f, symbol_f) | f \in F, symbol_f \in domain(f)\}$ that are currently known to the agent. As explained earlier in Section 4.2.2.1, ROSIE may begin with an incomplete knowledge of the domains of perceptual attributes. It learns the domain through instruction when the need arises.

The current state of the agent is composed of a set of beliefs about observable objects represented as relational predicates defined over visible objects. They are of following types.

- *Existence*, P_e. These are predicates that capture if the object is observable or not.
- *Category*, P_c. These predicates indicate if the an object o is a location or a movable block.
- *Spatial relations*, P_r. These are predicates that indicate if a spatial relation r exists between two objects, $o1$ and $o2$. The spatial relation is represented as a composition of primitive predicates in SVS as described in Section 4.2.2.2 and can be learned from instruction.

- *Functional state*, P_f. These predicates are available for certain objects and capture their function state such as if they are *open* or *closed* and *on* or *off*.

4.2.3 Interaction Cycle

We pre-encoded Rosie with rules that assist in parsing and categorizing utterances, maintain the interaction state, ground utterances to extra-linguistic knowledge, and assist learning. Below, we describe the interaction cycle implemented using these rules which forms the basis of processing in Rosie. During the interaction cycle, Rosie selects and applies operators to interpret the instructor's utterances and generate behavior. The interaction cycle begins with a natural language utterance from the instructor (shown using solid line in Figure 4.4) and is processed in following phases. Consider the example in Figure 4.5. An interaction cycle begins when the instructor commands Rosie to perform the *move* task in Line 2.

4.2.3.1 Lexical Processing

LG-Soar (Lonsdale et al., 2006), a natural language component implemented as operators in Soar, generates a syntactic parse of the utterance using a static dictionary and grammar. It uses part-of-speech tags to create a parse in Rosie's working memory, identifying the useful content in the message. This parse is further categorized as `task-command`, `goal-description`, `descriptive-sentence`, etc. based on its lexical structure. For example, Line 2 is characterized as a `task-command` in this phase.

4.2.3.2 Interaction Management

After the utterance has been categorized, Rosie interprets within the context of the ongoing dialog. Using the context of the ongoing interaction and the contents of instruction,

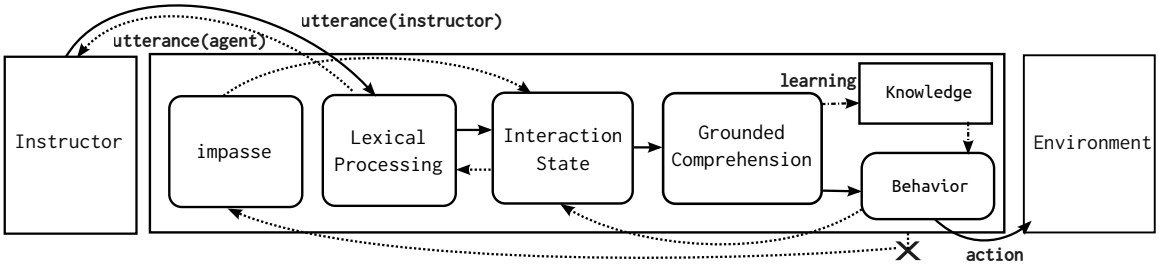

Figure 4.4: Rosie's Interaction Cycle

```
1.  Rosie: Give me a task.
2.  Instructor: Move the orange triangle to the pantry.
3.  Rosie: What kind of attribute is orange?
4.  Instructor: A color
5.  Rosie: (learns semantic categorization)
6.  Rosie: What is the goal of the action?
7.  Instructor: The goal is the orange triangle in the pantry.
8.  Rosie: I don't know the preposition in. Please teach me
..  with examples.
9.  Instructor: The orange triangle is in the garbage.
10. Rosie: Okay.
11. Rosie: (learns semantic composition of 'in')
12. Rosie: (learns goal of 'move')
13. Rosie: What action should I take next?
14. Instructor: Pick up the orange triangle.
15. Rosie: (picks up the orange triangle)
16. Rosie: What action should I take next?
17. Instructor: Put the triangle in the pantry.
18. Rosie: (puts the triangle in the pantry)
19. Rosie: (learns 'move' with retrospective projection)
```

Figure 4.5: Rosie's learning interaction trace.

Rosie creates a *purpose*[1] to pursue using pre-encoded heuristics. The purpose may include performing actions in the environment in response to commands from the instructor, providing responses to instructor's queries, or learning from provided demonstrations. The purpose of the instruction in line 2 is executing the *move* task with the relevant objects. Chapter 7 describes in further detail how the state of interaction is represented and maintained.

4.2.3.3 Situated Comprehension

To gain useful information from an instruction, Rosie must ground linguistic references to objects, spatial relationships, and actions. We use the term *map* for structures in semantic memory that encode how linguistic symbols (nouns/adjectives, spatial prepositions, and action verbs) are associated with perceptual symbols, spatial compositions, and task knowledge. Maps are learned through interactions with the environment and the instructor and are stored in semantic memory. To ground a sentence, an *indexing* process (Chapter 6) attempts to retrieve relevant maps from semantic memory so that it can connect the linguistic terms with their referents. If the terms are successfully mapped, Rosie uses constraints derived from the retrieved maps, the current state, known action models, and the interaction state to create a grounded representation of the instruction. These sources of knowledge can be used to resolve semantic ambiguity. If indexing fails to retrieve a map or there is insufficient knowledge to resolve the ambiguity, an impasse will arise (see Section 4.2.3.5).

In the example (in Figure 4.5), during comprehension of the instruction, Rosie fails at associating the adjective word *orange* to a color classifier. This results in an impasse and consequent interactions to learn this knowledge (see Section 4.2.3.5).

[1]We use the term *purpose* to refer to Rosie's internal reasoning goals in order to distinguish them from external task goals such as `(in(A1,pantry))`

4.2.3.4 Behavior

If Rosie is successful in generating a grounded representation of the instructor's utterance, it attempts to pursue the purpose of the utterance. A natural language command *Pick up the orange triangle* results in Rosie picking up the referenced object (Lines 14, 15 in Figure 4.5). Apart from executing tasks, Rosie can also generate linguistic descriptions of the scene and can be queried about various objects and spatial relationships to verify its learning. If Rosie does not know how to execute a task, an impasse will arise (see Section 4.2.3.5).

4.2.3.5 Impasse and Acquisition

If Rosie fails to ground an utterance or is unable to execute the requested task, an impasse arises. In response to the impasse, Rosie initiates a new interaction with the instructor (shown using dotted lines in Figure 4.4) to acquire the missing knowledge. If there are multiple failures during interpretation of a new instruction (comprehension of Line 2 in Figure 4.5 results in failure for the words *orange* and *move*), Rosie processes them one at a time, leading the instructor through a series of interactions until it can resolve all the impasses. From impasses arising during *situated comprehension phase* and the ensuing interactions (Lines 3, 4, and 5), Rosie can learn groundings for nouns, adjectives, and verbs. From impasses arising during *behavior* (Lines 13, 14, 15), it learns task representations (Chapter 8).

4.3 Summary and Discussion

In this chapter, we reviewed the Soar cognitive architecture and presented an analysis of how different components in Soar contribute to the design of Rosie. The various long-term memories and spatial-visual system are useful in representing different aspects of task and verb knowledge. Soar's commitment to online processing, memory access, and

learning along with architectural guarantees on real-time reactivity make it an ideal architecture for designing interactive agents. We gave a brief overview of Rosie and described the representations used for encoding perceptual and spatial knowledge. As demonstrated in Mohan, Mininger, Kirk, et al., 2012, this knowledge can be acquired from SII. For the rest of this thesis, we will assume that the agent has this knowledge from previous SII episodes. We also described the interaction cycles that forms the basis of SII. It serves to integrate various capabilities in Rosie from lexical processing to learning. A phase in the cycle produces results which are required by the next phase in the cycle. Failures in a phase (impasses) result in Rosie analyzing their cause and asking appropriate questions to resolve the issue. This tightly integrates processing with knowledge acquisition and biases learning to what Rosie requires for making progress on the task at hand.

Chapter 5
Verbs and Tasks

Describing an action (or an event) and its participants is a central aspect of any human language. In the English language, this function is largely served by the *verb* grammatical class, often aided by *prepositions*, *adverbs*, and *wh-clauses*. It, therefore, is critical for taskable communicative agents to represent, understand, and generate verbs in order to use language effectively. For an agent that uses language for collaboration and learning, it is essential to ground the lexical aspects of the verb and its arguments in internal representations of goals, intentions, tasks, and experience.

There has been extensive work in grounding concrete nouns and adjectives and prepositions for physically grounded agents. However, the problem of grounding verbs is relatively unexplored. Why is this so? Gentner (2006) argues that *"the noun class has the privilege of naming the highly cohesive bits of the world, whereas verbs and prepositions have the job of partitioning the leftovers - a diffuse set of largely relational components."* In other words, concrete nouns and adjectives refer to naturally bounded referents, perceptible in the world. In contrast, even very concrete verbs such as those that describe actions, stand for complex relationships between their objects, goals, and execution policy. Acquisition of verbs, therefore, is a significant challenge, where the learner must induce hypotheses about how the objects of the verb are related and how these relationships affect behavior.

The difference in difficulty of learning nouns versus verbs is also apparent in human language acquisition. Evidence from research in native language acquisition suggests that not only are verb words harder to learn than noun words (Gentner, 1982; Childers

and Tomasello, 2002), but also, generalizing verbs to novel scenarios poses a significant challenge to children and is harder than generalizing nouns to new classes of objects (Kersten and Smith, 2002; Imai et al., 2005). In extending a verb that has been mapped onto an action involving an object, the object must be separated from the action and be treated as a variable that can be changed. Additionally, verbs place some constraints on the types of arguments with which they can be used. Further, the use of verbs in human language is highly variable and often omits arguments. These aspects of verbs makes learning them especially challenging.

The general problem of grounding verbs is incredibly challenging as they encode a variety of things including actions (*bring*, *make*), occurrences (*happen*, *become*), or state of being (*exist*, *stand*). In this thesis, instead of studying the general problem of verb grounding, we focus on concrete verbs that describe tasks. We study how verbs such as *put* can be grounded in *learnable* task representations. This not only allows the agent to produce behavior in response to imperatives such as *put away that book*, but also allows a human user to teach new verbs and corresponding behaviors, thereby, extending agent's linguistic capability and functionality. From the perspective of SII agents, learning to align the lexical structure of verbs with task representations facilitates language-driven future collaboration with humans along with assisting in learning complex, hierarchical tasks.

By proposing a task-oriented representation of verbs, this chapter sets the stage for learning new tasks (Chapter 8) and comprehending imperatives (Chapter 6). We begin by a summary of prior work in Section 5.1. Then, in Section 5.2, we study how humans use task verbs for describing domestic chores. This analysis motivates the task-oriented representation for grounded verb semantics in Section 5.3.

5.1 Prior Work on Grounded Verb Semantics

The challenge of grounding verbs in actions has been addressed from different perspectives in AI community. Siskind (2001) presents LEONARD, an agent that recognizes and describes simple actions from a sequence of images. The lexical semantics of verbs are grounded in perceptual data and are represented as event-logic expressions describing the force-dynamic relations between the participants of the action. The representations are insensitive to specific motion profiles and the presence of irrelevant objects in the field view. This work pioneered the research in action/event recognition in computer vision and describes a powerful approach. However, the proposed representations do not encode the control knowledge required to perform an action in the environment which is essential for producing behavior in response to action commands.

Others have looked at grounding verbs in motor-control programs. Bailey (1997) introduced *executing-schemas* - a graphical Petri Nets representation for action control. Every verb is associated with an *executing-schema* which is designed to achieve a relevant goal (such as obtaining an object) and may encode various ways of achieving the goal based on the world state. Although this is sufficient for producing behavior in response to task commands, the schemas are hand-engineered and the author gives no account of how and if these representations can be learned.

Roy et al. (2003) propose grounding verbs in motor programs or procedures. For example, the verb *touch* can be grounded in a perceptually-guided program below.

```
procedure Touch(x) {
    repeat: Reach-towards(x)
    until touch sensor(s) activated
       if x in view then return success
       else return failure
       end if
}
```

This, too, does not provide an account of how these representations can be learned and relies on hand-engineered programs.

Recent work by Kollar et al. (2014) presented a system that can learn control programs corresponding to action command such as *follow the person to the kitchen* from human generated traces of similar behavior. The action command is represented as a set of *spatial description clauses* (SDCs) obtained by constraining factor graphs (Kollar et al., 2010). The SDC containing verbs are associated with motion trajectories from human traces. Given a command, the inference engine computes a set of states that must be achieved during motion based on human data. A controller then plans and executes a motion trajectory. The learning paradigm allows the system to handle some variation in verb usage. The learning paradigm is data intensive and relies on offline processing of human generated corpus. It is unclear if such representations can be effectively incorporated in interactive agents that learn online.

This thesis contributes a *mixed-modality* representation of task verbs that grounds lexical and semantic aspects of verbs in task goals and policy. The representation is *learnable* and can be acquired from human-agent instructional interaction.

5.2 An Analysis of Task Verbs

Although prior work has investigated a few representations for grounded verb semantics, none has characterized how verbs are used to describe tasks and if the proposed representations cover the variability in verb usage. We present a preliminary analysis of commonly used task verbs in a domestic or a kitchen environment below. This analysis has two components. The semantic analysis (in Section 5.2.2) is based on VerbNet (Schuler, 2005) and studies how humans use verbs to describe tasks while the task-oriented analysis (in Section 5.2.3) characterizes the goals and control structures of these tasks. This analysis motivates our representation (in Section 5.3) that incorporates linguistic, semantic, and procedural elements. It also serves as a useful tool to evaluate if the proposed representation and corresponding learning paradigm covers a variety of tasks (D12).

5.2.1 Chores Dataset

Chore lists are created by members of a household and are useful in managing tasks and the division of labor. They reveal the need, preferences, and the rules of the household. Cakmak and Takayama (2013) proposed using chore lists to guide the design of domestic robots as they give valuable information about what robotic capabilities will be useful for users. The language used in chore lists reflects how humans describe common household tasks, and therefore, is a good starting point for analyzing task verbs.

Cakmak and Takayama (2013) collected 25 chore lists from the world wide web by searching various combinations of the keywords *house*, *household*, *housekeeping*, and *task*, *chore*, *work*. We collected 20 additional chore lists for kitchen tasks by searching the world wide web with combinations of keywords *kitchen* and *tasks*, *chores*, *chore lists*. Following Cakmak and Takayama (2013), we pre-processed the data (henceforth, chores dataset) to separate out each task as a separate item. For example, the chore *make breakfast, lunch, dinner* was included as three items *make breakfast*, *make lunch*, *make dinner*. The lists were collated and were organized by the verbs used to describe tasks and then by their objects[1].

The kitchen chores dataset contains 53 verbs and the home chores dataset contains contains 46 verbs. The verbs are used with a variety of household objects (*kitchen chair*, *dishes*), locations (*curb*), appliances or instruments (*fridge*, *stove*), and surfaces (*tabletop*, *kitchen floor*).

5.2.2 Semantic Analysis

VerbNet (Schuler, 2005) is a lexicon that organizes and categorizes various English verbs on the basis of their common syntactic and semantic properties. It has been used for several linguistic tasks including human-agent dialog, verb sense disambiguation, and concept network creation. The representation of verbs in VerbNet captures both the syntax

[1]Available at http://www.shiwali.me/home-data.html, http://www.shiwali.me/kitchen-data.html

and semantics of the verbs and makes explicit links between the two. As the representation in VerbNet are declarative, it is insufficient for representing behavior or control policies. However, it identifies the classes of knowledge that are useful in linguistic tasks and therefore, should be included in grounded verb semantics. Below we present an analysis of verbs in the chores data set based on the components of semantic representation in VerbNet. Later in Chapter 6, we describe how some components of this knowledge are useful in understanding and executing task commands (or *imperatives*).

5.2.2.1 Thematic roles

Thematic roles (Jackendoff, 1972) express the role the object referred to by a noun phrase plays with respect to the action or state described by the sentence's verb. In the sentence,

> Tom broke the cup.

Tom serves the role of an *agent* - an active instigator of an event; and *the cup* serves the role of a *patient* - a participant undergoing a state change through an action. VerbNet's argument list consists of a set of 23 thematic roles that cover arguments for all verb classes. The chores dataset contains only a few thematic roles which are described below. The distribution is shown in Table 5.1.

- **Theme**: used for participants in a location or undergoing a change of location. Example: *Put the dishes in the dishwasher.*
- **Patient**: used for participants undergoing a change in state. Example: *Cook rice.*
- **Instrument**: used for objects (or forces) that come in contact with an object and cause some change in them. Example: *Cook potatoes on the stove.*
- **Spatial Locations**[2]:

[2]There is some ambiguity in VerbNet about classifying arguments as *destination* or a general *location.*

Thematic Role	Objects	Verbs
patient	dish, fridge, shower, sink, pans, meal, food	wash, clean, cook, fold
theme	dishes, napkins, placemats, garbage, cup, plate, clothes, laundry	put, set, take, clear, load, unload, carry
product	salad, dinner, meals, grocery list	make, prepare
instrument	soap and water, sponge, stove, non-toxic cleanser	clean, wipe, cook
destination	dishwasher, table, curb, street, counter	put, take, load, unload
source	dishwasher	unload, put
location	counter, dishwasher, freezer, table, pantry, shelf	empty, organize, set, clear

Table 5.1: Thematic roles represented in the chores dataset

  - **Destination**: end point of motion, usually introduced by a *to* prepositional phrase. Example: *Take the recycling to the curb*.
  - **Source**: start point of motion, usually introduced by a source prepositional phrase. Example: *Unload silverware from the dishwasher*.
  - **Location**: unspecified source, destination, or place. Example: *Clean the table*.

- **Product**: end result of a transformation, usually used with verbs of creation. Example: *Make a salad*.
- **Time, frequency**[3]: when should the task be done. Verbs *clean*, and *wipe* also contained arguments such as *after every meal*, *weekly* that identify when and how frequently the task should be done. These types or arguments are not represented in VerbNet.

Levin (1993) reports that for a wide range of activity verbs, the object may be left unexpressed. This alternation goes by several names including *unexpressed object*, *indefinite object*, and *indefinite NP deletion* alternation. The author only reports instances where the

[3]These roles are not included in VerbNet.

direct object is left unexpressed. However, in the chores dataset, it is apparent that other objects may also be left unexpressed.

- **Theme**. Verbs such as *load*, *unload*, *empty*, *clear*, *set* optionally leave their *themes* unexpressed.

    1. Unload the silverware from the dishwasher.
    2. Unload utensils from the dishwasher.
    3. Unload the dishwasher.

- **Location (destination)**. Verbs such as *take*, *put*, *load*, *carry* often leave the *locations* unexpressed.

    1. Take recycling to the curb.
    2. Take out recycling.

- **Instrument**. Verbs such as *clean*, *cook*, *wipe* leave the *instrument* unexpressed.

    1. Cook potatoes on the stove.
    2. Cook rice.

5.2.2.2 Selectional restrictions

Selectional restrictions indicate the constraints the verb imposes on the nature of the arguments it may be combined with. Usually, they are represented as labels (*edible*, *physical object*) which are obtained from a general ontology of objects in the world. For example, the verb *eat* takes *edible* as a direct object. VerbNet's ontology is hierarchically composed of *is-a* relationship with multiple inheritances. VerbNet does not make any distinctions between different types of selectional restrictions and only represents ontology-driven restrictions. However, research on human sentence processing suggests that some selectional restrictions may be non-linguistic in nature.

- **Semantic Knowledge**: Kamide et al. (2003) conducted a visual word experiment with a scene portraying a hare, a cabbage, a fox, and a tree and the auditory versions of the following sentences.

  The hare will shortly eat the cabbage.
  The hare will be shortly eaten by the fox.

  They report that the second noun phrase was predicted (indicated by a saccade to the relevant object in the scene) before the onset of the referring noun. This suggest that humans rapidly integrate semantic domain knowledge (knowledge of the food hierarchy in this case) during comprehension. A general ontology of objects is insufficient to produce this behavior.

  In the chores dataset, this restriction plays a role when certain arguments are left unexpressed in a sentence. Examples:

  Put away books. (on the shelf).
  Put away leftovers. (in the fridge).

  The main verb is common in both sentences. However, the understood *location* is different and depends on the *theme*.

- **Environmental State**: Chambers et al. (2004) conducted experiments in which the participants were asked to follow instructions containing syntactic ambiguities (such as *pour the egg in the bowl over the flour*). The authors varied the affordances of task-relevant objects with respect to the action required by the instruction (whether one or both eggs in the workspace were in liquid form, allowing them to be poured). The number of candidate objects that could afford the action was found to determine whether listeners initially misinterpreted the ambiguous phrase (*in the bowl*) as specifying a *location*. The findings indicate that comprehension is influenced by the hearer's evaluation of how to achieve the goal in the current state. This re-

striction is not observed in the chores dataset because this depends on the state in which the hearer was asked to perform the task while the chores dataset only captures language. However, encoding this is necessary to match a human speaker's expectations.

5.2.2.3 Syntactic Frames

Syntactic frames are surface realizations for a verb. Example:

Agent v Patient
(Bob hits the ball.)

Frames describe constructions such as transitive, intransitive, prepositional phrases, etc. A syntactic frame consists of the thematic role in their argument position around the verb, the verb, and other supporting lexical items. Additional restrictions may be included based on number agreement and syntactical restrictions. Table 5.2 summarizes the common syntactic frames in the chores dataset.

Frame	**Verbs**
V Patient	clean, wash, wipe, dry, cook
V Patient *with* Instrument	wipe
V Theme *in, on, to* Destination	put, take, load, carry
V Theme *from* Source	put, unload
V Location	empty
V Theme *from* Location	empty, clear
V Theme	set
V Product	make, prepare

Table 5.2: Syntactic frames represented in the chores dataset

5.2.2.4 Semantic Predicates

Semantic predicates encode relational information about the objects of the verb including their state, spatial configuration, movement, manner, and time. VerbNet specifies whether a predicate is true at all times in the event *(E)*, at the start *(start(E))*, in the preparatory

(during(E)), culmination *(end(E))*, or consequent *(result(E))* stage of the event. An example from VerbNet is below.

> *Paula hit the ball.*
> (Agent V Patient)
>
> ```
> cause (Agent, E)
> manner (during(E), directedMotion, Agent)
> !contact (during(E), Agent, Patient)
> manner (end(E), forceful, Agent)
> contact (end(E), Agent, Patient)
> ```

Such knowledge is useful for extra-linguistic reasoning about the linguistic input and potentially in producing behavior. However, this knowledge in VerbNet is not grounded and is incomplete. Even though VerbNet encodes the truth values of predicates with respect to the action denoted by the verb, it does not contain any knowledge about how the action can be performed or how the predicates connect to what is observed in the world.

5.2.3 Task-Oriented Analysis

We now analyze the various tasks that are represented in the chores dataset. This analysis informs how the task representation should be structured and what components should be included in the agent such that it can be used to encode several tasks.

5.2.3.1 Skill-Oriented versus Goal Oriented

Cakmak and Takayama (2013) report that in the home chores dataset, cleaning tasks comprise a significant number of chores and involve verbs such as *clean*, *wipe*, *vacuum*, *dust*, *wash*, *sweep*, *mop*, and *scrub*. Similarly, in the kitchen chores dataset verbs *clean*, *wipe*, *wash*, *mop* and *sweep* indicate cleaning chores. The *patients* include rooms, floors, appliances, furniture, and parts of the house including cabinets, sinks, windows, and countertops. The cleaning tasks can be characterized as a *skill* of applying a tool (such as *sponge*,

usually an unexpressed object) on the surface specified by the *patient* of the verb. Such skills can be represented as continuous policies.

Other tasks are *goal-oriented* and can be characterized as applying several continuous policies in succession to achieve a goal state in the environment. These tasks can be broadly categorized as -

- **Organizational**. These involve moving and rearranging household items where the final goal can be characterized as establishing a set of spatial relationships between items. Common verbs include *organize*, *put-away*, *pick-up*, *tidy*, *empty*, *clear*, *load*, *unload*, *clean* (pantry).

- **Functional**. These involve acting to change the state of objects. Common verbs include *cook*, *make*, *wash* (dishes, clothes), *prepare*.

5.2.3.2 Implicit Information

- **Implicit Goal**. In the chores dataset, the goals of tasks are never explicitly identified. Instead, the verbs used to describe them serve as abstractions over their goals and how the tasks are executed.

  > Example:
  > *Take the trash to the curb.*
  > does not specify what spatial relations should be established between participant objects.

- **Implicit Arguments**. Semantic analysis of several task verbs reveals that occasionally, some thematic roles are left unexpressed in a task command. This causes critical task arguments to be left unspecified,

  > Example:
  > *put-away the toys*
  > *unspecified location*: does not specify where the toys must be placed

or implicitly identified.

> Example:
> *set the table*
> *implicit theme*: does not identify which items should be placed on the table.

5.2.3.3 Goal Types

- **Achievement**. A large majority of tasks can be characterized as *achievement* tasks. The agent must undertake a sequence of actions to achieve the task goal. A few tasks were formulated as *prospective* goals and the descriptions included information about when the tasks should be done. Example:

  > *Clean the dishes after every meal.*

- **Maintenance**. Maintenance tasks require the agent to observe and maintain some world state defined by the goal. Only three instances of *maintenance* tasks were observed, all of which involved the use of verb *keep*. For example:

  > *Keep the counter-tops clean.*

- **Performance**. Tasks such as *patrol* refer to performing an activity instead of acting to achieve a goal. No instances of such tasks were observed in the chores dataset.

5.3 A Task-Oriented Representation for Grounded Verb Semantics

The analysis above informs the formulation of representations that are useful not only in comprehending language but also for encoding behavior. The representation should have two characteristics. First, it should be *comprehensive* - apart from encoding the syntactic information, it must encompass the semantics useful for understanding language and the

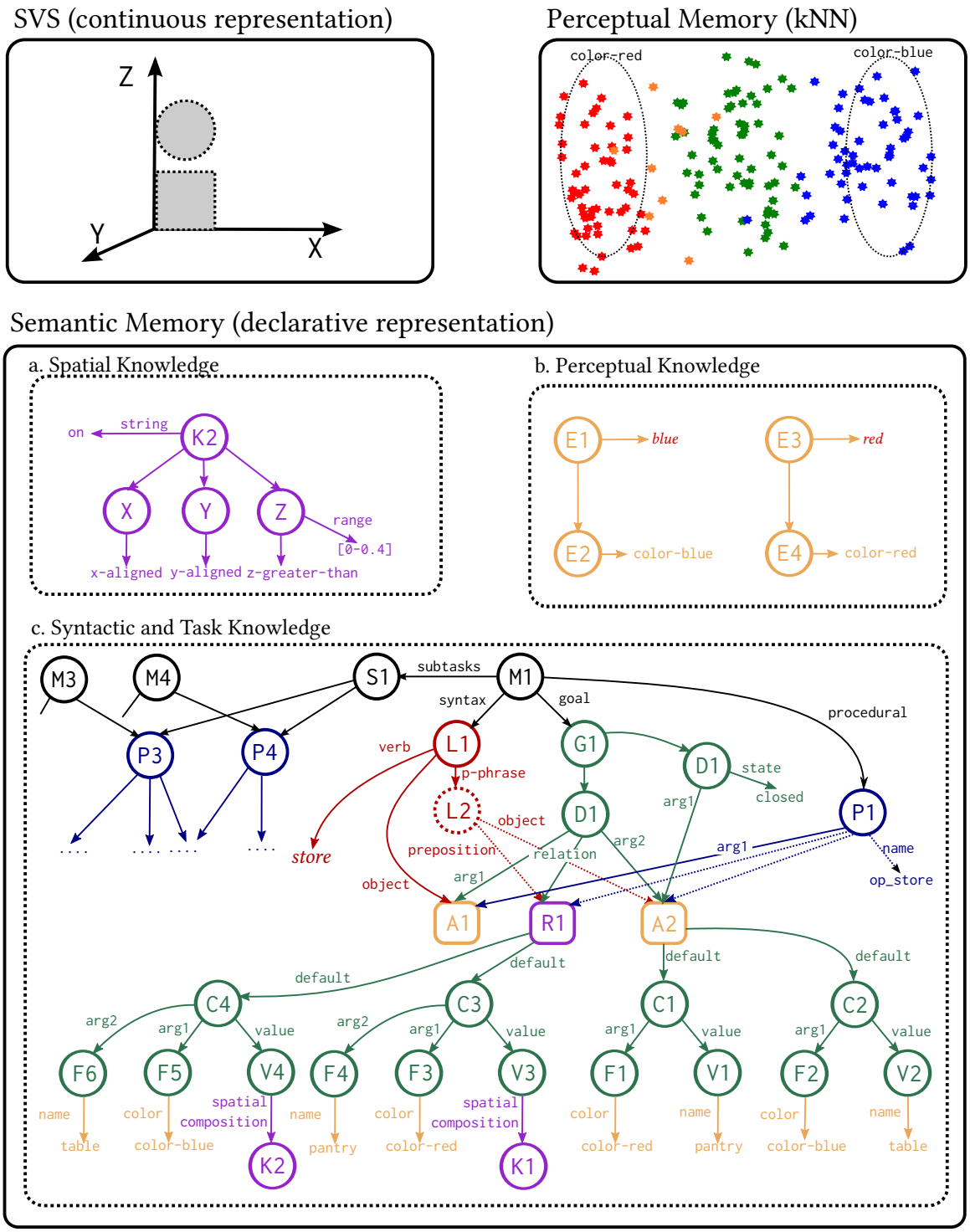

Figure 5.1: A task-oriented representation for grounded verb semantics in Rosie's memories. Nodes and symbols in color purple correspond to spatial knowledge, orange to perceptual knowledge, green to goal definition and constraints, blue to procedural knowledge, and red to syntactical knowledge.

knowledge useful in task structuring and execution, establishing explicit links between the two. Furthermore, if the agent is functioning in real-world-like scenarios, the representation must handle continuous and probabilistic properties of its environment. Second, the representation must be *learnable* - the agent must be able to induce this representation from its experience of its domain and interactions with a human instructor.

We now propose a *mixed* representation that employs modality-specific representations for various aspects of task verbs. Chapter 8 addresses interactive methods for learning these representations and Chapter 6 discusses the contributions made by this representation in comprehension of action commands. We refer to Figure 5.1 throughout this section to explain different components of our representation. The graph in block c in Figure 5.1 shows a *task concept network* (TCN) that mediates various modality specific representations. The symbols in TCN serve as pointers to knowledge in different long-term memories and SVS. For example, the symbol `color-blue` refers to a class in the *kNN* classifier in perceptual memory and the symbol `x-alignment` refers to alignment along the `X-axis` in SVS. The modality specific knowledge is accessed and processed when the need arises. For example, for executing the task *put the red object on the table*, Rosie needs to find a point on the scene that satisfies an `on` relationship with the *table*. SVS will compute this point using the definition of *on* (graph in block a) and grounding the symbols in its knowledge of alignments, containment, etc. in the continuous environment.

The figure shows the TCN that corresponds to the verb *store* which can be used in two different alternations.

a. *Store the red cylinder on the table.*
b. *Store away the red cylinder.*

5.3.1 Syntactic Knowledge

Syntactic knowledge of a verb is represented declaratively in Rosie's semantic memory. An example is shown in Figure 5.1 as the red colored subgraph rooted at node `L1`. The

subgraph encodes that the verb *store* has a direct object and optionally (shown in dotted lines) a prepositional phrase that identifies the location. This knowledge serves a function similar to syntactic frames in VerbNet. Thematic roles are not explicitly encoded but are implicitly represented in the task goals and policy (explained in later sections).

5.3.2 Association Knowledge: Map

We use the term *map* for nodes and edges in semantic memory that encode how linguistic symbols (nouns/adjectives, spatial prepositions, and verbs) and constructions are associated with perceptual symbols, spatial compositions, and task knowledge. For comprehending a word (explained in detail in Chapter 6), Rosie searches its semantic memory for a corresponding *map* that will allow it to access relevant non-linguistic knowledge implementing referential comprehension (desiderata D2).

Nodes `M1`, `A1`, `A2`, `R1` align the syntax of the verb with the structure of the task (described in Section 5.3.3) and corresponding p knowledge (described in Section 5.3.4). `M1` is is an abstraction over all knowledge that corresponds to the verb *store*. Nodes `A1` and `A2` are slot nodes that can be filled with physical objects referred to noun phrase (NP) object of the verb. The slot node `R1` can be filled by known spatial compositions. The slot nodes constrain the instantiation of structural and procedural knowledge to objects and spatial compositions referred to by the task command or known default instantiations (described later).

5.3.3 Structural Knowledge

The structural knowledge is encoded declaratively in Rosie's semantic memory and consists of the following components.

5.3.3.1 Goal Description

An explicit definition of goal is stored in semantic memory. The goal definition is encoded as a composition of spatial relation or functional state predicates. The spatial relations are directly grounded in positional data in perceptions. The functional state is grounded in virtual state of objects in the environment.

The subgraph rooted at `G1` encodes the goal definition for the *store* verb. It is composed of two predicates: one that identifies the spatial relation that is to be established between the object and the location and the other that established the state of the location. The goal instantiation is constrained to the values in slot nodes `A1`, `R1`, `A2`. If a slot is filled by referents grounded in the environment, they are used for goal instantiation. Otherwise, their associative default values are used (described below).

5.3.3.2 Associative Default Values

As noted earlier, several verbs have unexpressed object alternations. In *put away the books*, the role of *location* is left unexpressed. Usually, the implicit or unexpressed arguments depend on the specified object and its semantic or perceptual categorization. For the task command *put away the book*, the implied location is the *shelf* but for *put away the clothes*, the implied location can be a *closet*. The associative default values represents such associations between the specified object and unexpressed objects of the verb.

The subgraph rooted at `C1` gives an example. It encodes that in situations where the value of slot `A2` is not constrained linguistically, if `arg1` of predicate `D1` contains the perceptual feature `color-red` (that corresponds to a class in the *kNN* classifier), then fill `A2` with an object that contains the feature `pantry`. Similarly, the subgraph rooted at `C2` provides an alternative location if the specified object contains the perceptual feature `color-blue`.

5.3.3.3 Subtasks

The subgraph rooted at S1 establishes connections between the task and its constituents subtasks (P3, P4, ...).

5.3.4 Procedural Knowledge

Task execution knowledge is encoded in ROSIE's procedural memory as rules. This knowledge is useful in producing behavior in response to task commands from the instructor. It is encapsulated in a task operator which is proposed, selected, and applied in appropriate environmental states. The instantiation of the task operator is dependent on how the task command is grounded in the environment. The edges P1,A1, P1,R1, and P1,A2 align the procedural knowledge of *store* encapsulated in the task operator op_store with structural and linguistic knowledge of *put*. In the following sections, we describe the components of task execution knowledge.

As explained earlier, the beliefs about the current state $s \in S$ are encoded as a set of relational predicates P defined over the set of objects O. The execution knowledge for a task is encapsulated in a task operator ($t \in T$, op_store in this case). The execution knowledge has the following components.

5.3.4.1 Parameters

Every task operator t is instantiated with a set of objects ($O^t \subseteq O$) that are involved in the application of the task. Parameters can be explicit or implicit.

- Explicit parameters are the objects of the verb in the task command. In *store the red cylinder*, the object described by *the red cylinder* is an explicit parameter.
- Implicit parameters are objects that are left unexpressed but are critical for task execution. The locations *pantry* and *table* are implicit parameters of op_store and are elaborated during goal elaboration described below.

5.3.4.2 Goal Elaboration

A goal elaboration operator is the proceduralized instance of the goal definition. It augments the task operator with instantiated goal predicates. The application of this operator also elaborates the implicit parameters of the task. If the implicit parameter are constrained to different values based on the attributes of explicit parameters, those constrains are included as well.

For the *store* task, the goal elaboration operator will instantiate the goal as `(o1, in, pantry)` and `closed(pantry)`, if `o1` is red in color and as `(o1, on, table)`, if `o1` is blue in color.

5.3.4.3 Availability Conditions

An important aspect of knowing a task is knowing when that task is available to be or should be executed in the environment. In Rosie, the availability conditions are encoded as operator proposal rules. These rules propose task operators (`op_store` in Figure 5.1) which may later be selected for execution. The proposal rule incorporates the following -

- **Selectional restrictions**. The rules incorporate the affordance of objects relevant to the task. Only those task operators are proposed that are afforded in the current state of objects and their configuration. For example, the proposal rule for `op_store` tests if the object is *movable* and is *clear*. The availability of certain tasks biases resolution of referring expressions in case of ambiguity (further explained in Chapter 6). This implements environmental state driven selectional restrictions of verbs (5.2.2.2).

- **Goal sensitivity**. Acquired availability conditions also test for the absence of the goal realization in the current state. For example, the proposal rule for `op_store` test if the object is not already in its desired location.

- **Prospective goals**. It was noted earlier that some chore descriptions described when

is it appropriate to perform the chores (*clean the dishes after every meal*). This information can be encoded in the proposal conditions, triggering the task operator when those conditions are met in the environment. We have not addressed this capability in this thesis but will explore it in the future.

5.3.4.4 Subtask Availability Conditions

These rules are similar to task availability rules but they also include the task context including its name and its parameters. These rules propose the subtasks of t while executing it.

For *store*, the subtask availability conditions propose `op_put-down(o1, [loc])` operators to various locations, if the robot is holding `o1`.

5.3.4.5 Policy

The policy rules select the subtasks based on the current state of the environment while executing the tasks.

For *put*, the policy rules select the subtask operator `op_place(o1, in, pantry)`, if the *pantry* is open.

5.3.4.6 Application

These rules apply the task and its substasks to the current state. These rules may affect the state by either executing the motor commands in the environment, or through a deliberate step in internal reasoning. The application rules determine how the objects in the environment will be manipulated, implementing the thematic roles implicitly. For example, in response to *place the red cylinder in the pantry*, the `place` operator causes the robotic arm to use its gripper to pick the *red cylinder* up and place it at a different co-ordinate in the *pantry*. In this case, the object *red cylinder* serves the role of *theme* of the verb *place*.

5.3.4.7 Model

The model predicts the future state s' of the world after the task is executed in the current state s. It is represented as a set of rules that encode how predicates in s transition to predicates in s' under task t. This is useful in exploring a path to the goal in addition to retrospectively explaining the instructions given to execute a task.

The model the operator `pick-up(o1)` encodes that the resulting state will include the predicate `holding(o1)`.

5.3.4.8 Termination Conditions

This rule detects if the goal predicates are realized in the environment and terminates the task policy. For *store*, this rule will monitor if the predicates `(o1,in,pantry)` and `closed(pantry)` are realized in the environment and will terminate the policy if they do.

5.4 Discussion

In order to make progress toward interactive agents that can communicate about activities and tasks, it is important to develop representations that align the syntactic and semantic aspects of verbs with knowledge of executing activities and tasks. This chapter presents a preliminary analysis of verbs commonly used to describe household chores. A typical chore description consists of a relevant verb along with a direct object and may optionally include prepositional objects. The analysis reveals some complexities in the use of verbs. Some objects that are critical to verb understanding and task execution may be left unexpressed. Another critical aspect of verbs is semantic and state-driven selectional restrictions it imposes on the nature of its arguments. Prior work in grounded language understanding and learning has not discussed such issues that lie on the interface of task execution and verb semantics. A major contribution of this thesis is a learnable verb representation that not only encodes task knowledge but also uses this knowledge to address

variations in verb usage. Although, the analysis and our representation only scratches the surface of complexity in verb usage in natural language, it takes a step toward developing a comprehensive theory of grounded verb semantics.

A concern about the representation proposed here is that it is symbolic and discrete, and therefore, unsuitable for behavior in the real-world. However, the representation is hybrid and incorporates probabilistic and continuous information in addition to discrete symbols. While some symbols such as those that describe the goal, are abstract, others are concrete and are grounded in sensory data and motor actuation policies. Perceptual symbols correspond to classes in Gaussian kNN classifiers and spatial symbols correspond to alignment of objects in the continuous space and distribution over distances between objects. Abstract symbols such as goals of tasks are defined as compositions of such grounded symbols and consequently encode non-discrete information. Similarly, policy symbols that define behavior eventually ground out to continuous motor-control programs. The symbols in the representation serve as pointers to a variety of knowledge types, often represented in other memories. This knowledge is accessed when needed to execute a task. The proposed representation is also relational and can be used with knowledge-intensive learning algorithms such as explanation-based generalization Mitchell et al., 1986 resulting in quick generalization and learning from small data online (described in Chapter 8).

5.5 Looking Ahead

This chapter proposes a mixed-modality representation which encompasses lexical, semantic, and task-oriented aspects of verbs. The representations are useful in linguistic tasks and are learnable. In Chapter 6, we propose an Indexical Model for comprehension that uses these representations for reducing ambiguities in generating grounded representations of task commands (or imperative sentences). In Chapter 8, we propose an

interactive learning paradigm based on explanation-based generalization (EBG: Mitchell et al., 1986; J. Laird et al., 1986) that can acquire these representations through SII.

Chapter 6
Comprehending Situated Commands

Communication in collaborative task execution (or SII) is *situated*. The speaker's (instructor's) linguistic utterances refer to objects, spatial configurations, and tasks in the shared environment. To respond and react to utterances, the hearer (learner) must associate the amodal linguistic symbols (*words*) and constructions (*phrases*) with the modal representations of perceptions, environmental state, domain knowledge, goals, and policies that are required for reasoning about and manipulating the environment.

Being situated provides a common ground of shared perceptions, goals, and domain knowledge that can be exploited during linguistic communication. Information that is apparent from the current state of the environment or that is a component of shared beliefs can be left out of the linguistic utterance by the speaker. This results in more efficient (fewer words) but ambiguous utterances. Humans frequently use referring expressions such as *it* or *that cylinder* that do not by themselves provide enough discriminative information for unambiguous resolution. The speaker assumes that the hearer can exploit extra-linguistic information, such as the context of the ongoing discourse or the state of current task execution for unambiguous comprehension. Similarly, several verbs such as those in the chores dataset studied in Chapter 5 are used in alternations where their argument are left unexpressed. Imperative sentences such as *take out the trash* incompletely specify the task by omitting critical information about the location where the *trash* should be moved. Such ambiguities and implied information make situated comprehension a significant challenge for interactive agents.

This chapter studies how imperative sentences such as those in the chores dataset can be comprehended by grounding them in perceptual, spatial, and task knowledge. In the following sections, we introduce the problem of comprehending imperative sentences (Section 6.1), give an overview of the Indexical Hypothesis (Glenberg and Robertson, 1999) - a psycholinguistic theory of grounded comprehension (Section 6.2), present the Indexical Model (Mohan, Mininger, and J. Laird, 2014) - an implemented computational model based on the Indexical Hypothesis (Section 6.4), describe how complexities such as the use of ambiguous referring expressions (Section 6.5) and unexpressed object verb alternations (Section 6.6) can be addressed by using non-linguistic context and knowledge, and discuss the degree to which the Indexical Model satisfies the requirements of SII agents (Section 6.7).

6.1 Comprehending Imperative Sentences

In a joint collaborative activity, imperative sentences convey that the speaker intends the hearer to complete a task. The joint communicative goal is for the hearer to identify the intended task and relevant objects and to correctly instantiate the task goals. A typical imperative sentence in our domain is composed of a verb that indicates an action/task to be taken in the environment which is then instantiated with objects described with noun phrases (NPs). An example is - *move the red large block in the pantry*. The goal of comprehension of imperative sentences is to generate an instantiated task representation that includes the task goals and execute the corresponding policy.

An example is shown below

> **Instruction**: *Move the red, large, triangle to the right of the blue cylinder.*
> **Interpretation**: `task: operator op_move, arg1 O1, arg2 O2, desired.predicate D1: O1, P4, O2.`

where the object `O1` satisfies the descriptive referring expression (RE) *the red, large,*

triangle, the location `O2` satisfies the descriptive RE, *the pantry*, and the `D1` corresponds to the spatial predicate composed of objects `O1`, `O2`, and spatial composition `P4` associated with *to the right of*. Operator `op_move` is an abstraction over the execution policy corresponding to the verb *move*.

There are several challenges in generating a grounded task instantiation. A few of them arise because natural language is ambiguous. Lexical knowledge and grammar of the language may be useful in rejecting several possible task instantiations that can be mapped to the language, however, typically such knowledge under-constrains the space of potential instantiations. Evidence from cognitive linguistics (Piantadosi et al., 2012) suggests that ambiguity in language is useful in encoding a large amount of knowledge with fewer symbols. The extra-linguistic contexts these symbols are used in, provides the remaining constraints to generate an unambiguous representation. Both the speaker and hearer are aware of the extra-linguistic context of language and use it to generate meaning. Some challenges are described below.

- *Referring expressions*: The use of referring expressions to indicate the object of interest is context dependent. Consider the RE *red large triangle* and the object `O1` it refers to in example above. Consider an environmental state in which the other object is smaller than `O1`. The instructor may refer to `O1` as the *large object* and omit information about color (*red*) or shape (*triangle*). Other contexts such as the context of the current task, attention, affordances of action may cause the instructor to use different REs including the severely under-constraining pronoun *it* to refer to the same object. The comprehension model in the agent should be robust to such context sensitive use of REs.

- *Unexpressed object alternations of verbs*: As discussed in Chapter 5, several verbs can be used in alternations that leave critical task arguments unexpressed. Moreover, the goal and task structure corresponding to the verb may be different based on the semantic and perceptual categorization of its objects. Linguistic information by

itself does not provide enough information to correctly instantiate a task. Therefore, the agent may have to rely on its knowledge and experience of its domain to generate instantiations when the linguistic information falls short.

- *Verb and task argument alignment*: Consider the sentence in the example above. Successful execution of the task described by the sentence requires the agent to correctly instantiate the spatial predicate `D1` such that it is in agreement with the argument structure of the verb *move* (`p1(O1, P4, O2)` instead of `D1(O2, P4, O1)`). This constraint is not explicit in the linguistic structure of the sentence and has to be derived from the domain models of the environment and prior experience with the verb.

- *Preposition phrase attachment*: The decision regarding the site of attaching the preposition phrase is ambiguous in English language. For example, the sentence, *move the red large object in the pantry to the right of the green cylinder*, is ambiguous. It could mean either *move (the red large object) (to the right of the green cylinder in the pantry)*, or *move (the red large object to the right of the green cylinder) (in the pantry)*. Such ambiguities can be resolved by analyzing the current situation and reasoning about the likely interpretation.

6.2 The Indexical Hypothesis

The Indexical Hypothesis (Glenberg and Robertson, 1999) of language comprehension explains how sentences become meaningful through grounding their interpretation in situated action. The hypothesis asserts that comprehending a sentence requires three processes:

- *indexing* words and phrases to referents that establishes the contents of the linguistic input,

- *deriving* affordances from these referents,
- and *meshing* these affordances under the guidance of physical constraints along with the constraints provided by the syntax of the sentence.

According to the hypothesis, the linguistic information specifies the situation by identifying which components (objects, relationships, etc.) are relevant, and the semantic and experiential knowledge associated with these components augments the linguistic input with details that are required for reasoning and taking action. In this formulation of sentence comprehension, linguistic symbols (words) and constructions (grammatical units) serve as cues to the hearer to search the common ground which may involve shared perceptions, experience, or common sense knowledge in order to identify the referents. The referents are then composed together to generate the meaning of a sentence.

6.3 Our Approach

Earlier work on the Indexical Hypothesis of language comprehension identifies the processes that humans use for comprehension (Glenberg and Robertson, 1999) and provides supporting data from human studies (Kaschak and Glenberg, 2000). It, however, does not propose a computational model nor identify the representations and the computation necessary for implementing indexical comprehension. A contribution of our work is a computational model - the Indexical Model - that precisely defines the processes described in the Indexical Hypothesis. It pursues the primary thesis that language comprehension can be formulated as a search over elements in perceptions, short-term memory, and long-term knowledge and composition of the relevant elements under syntactical and physical constraints. We also propose methods for referring expression resolution and handling unexpressed argument alternation of verbs within the Indexical Model. The model is implemented in Soar and has been integrated with the task learning component described in Chapter 8.

We show that our formulation of situated comprehension addresses several desiderata for SII identified previously in Chapter 2 and reiterated below.

- *Referential* (D2). It translates linguistic symbols to symbols in perceptions, spatial reasoning, and task representation to produce behavior when given an imperative sentence.

- *Expandable* (D5) and *Active* (D4). If there are failures in comprehension because Rosie lacks relevant knowledge, the Indexical Model provides information that is useful in learning the lacking knowledge through instruction. As Rosie learns online about its environment, the Indexical Model can use this knowledge for language comprehension. Prior approaches (Liang et al., 2009; D. Chen and Mooney, 2011; Tellex et al., 2011; Matuszek et al., 2012) on situated comprehension incorporate batch-learning methods that make a distinction between the training phase and the trial phase. In these approaches, the trained model cannot be extended online to comprehend new sentences. In comparison, the Rosie reverts to learning if it lacks knowledge to ground new sentence.

- *Integrative* (D3). Formulation of language comprehension as a search has a natural role for non-linguistic context. It guides search and provides additional constraints over hypotheses about the meaning of an ambiguous utterance. The Indexical Model can leverage perceptual, spatial, and task knowledge along with the context of the ongoing dialog for resolving ambiguous referring expressions. The non-linguistic context can also provide additional information to comprehension. The Indexical Model leverages task execution in situations where the certain objects are left unexpressed but are relevant to the task. Such questions that lie at the interface of language comprehension and behavior have not been studied in prior computational work on situated comprehension but are critical to robust communication. We demonstrate that diverse contexts can be exploited in the Indexical Model to re-

solve ambiguities arising from under-specific referring expressions and unexpressed argument alternation of verbs.

6.4 The Indexical Model of Comprehension

We assume that an imperative sentence consists of a verb, a direct object referring expression (RE), and optionally, a prepositional phrase that includes a preposition and an object RE. The process of indexical comprehension involves identifying the referents of the linguistic input (REs, prepositions, verbs) and composing them to generate an action instantiation grounded in the modal symbols that support reasoning about and manipulation of the environment.

Referents in the Indexical Model are derived from the following two sources. We refer to Figure 6.1 to explain the representation. The figure shows how referents are represented in ROSIE's semantic memory.

- *Working memory.* The working memory captures the current perceptual state of the world. As described earlier, an object in the ROSIE's view is described as a set of perceptual features (*color*, *shape*, *size*, *volume*) and their value assignments. As described earlier (in Chapter 4), an object $o \in O$ can be described as the set of feature-value pairs, $o = \{(f, symbol_f) | f \in F, symbol_f \in domain(f)\}$ that are currently known to ROSIE. $symbol_f$ correspond to a class in the kNN classifier associated with the feature f. O provides referents for RE in the imperative sentence.

- *Semantic memory.* Semantic memory contains ROSIE's long-term knowledge of the world and consists of the following -

  - *Perceptual classes*: symbols referring to kNN classes $symbol_f$ and the corresponding feature f (`C22` and `color` in network B in Figure 6.1). These provide referents for nouns and adjectives in the RE.

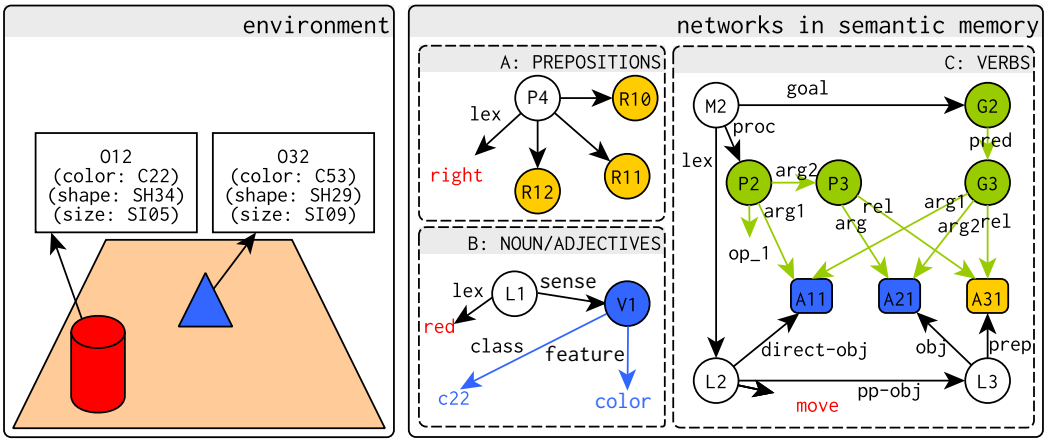

Figure 6.1: Environment state and the knowledge encoded in Rosie's semantic memory. The white nodes (`P4`, `L1`, `M2`, `L2`, `L3`) represent indexical maps between amodal linguistic symbols (in red: `right`, `red`, `move`) and modal domain knowledge. Yellow nodes (`R10`, `R11`, `R12`, `A31`) represent spatial symbols and slots (round rectangles: `A31`), blue nodes (`V1`, `A11`, `A21`) represent visual symbol and slots, and green nodes (`P2`, `P3`, `G2`, `G3`) represent procedural symbols.

- *Spatial compositions.* spatial compositions ($s \in S$) that describe alignment along axes and distribution over distances between objects in example instances (node `P4` in network A in Figure 6.1). These provide referents for prepositions.
- *Task policy and goal.* task execution knowledge (defined previously in Chapter 5 and shown in Network C in Figure 6.1). This knowledge provides referents for verbs.

Additionally, the semantic memory also contains *Indexical maps* (or *maps*) that are nodes (`L1`, `V1`) that align lexical symbols (*red*) with their corresponding referents (`C22`).

The Indexical Model uses a simple referential grammar: nouns and adjectives refer to visual properties; referring expressions refer to objects; prepositions refer to spatial relationships; verbs refer to task operators; and the imperative sentence refers to an task

operator instantiated with relevant arguments.

Consider the imperative sentence *move the large red cylinder to the right of the blue triangle*. Following the Indexical Hypothesis, comprehension is carried out as described in Algorithm 1. Some relevant terminology is below.

- The function $\textbf{smem-query}([a : b])$ queries the semantic memory for a long-term identifier (LTI) that contains the attribute-value pair $[a : b]$ and retrieves the graph rooted at the LTI to the working memory.
- n.a1.a2 refers to the node or symbol along edges a1 and a2 from node n.

The comprehension is carried out in following three stages in the Indexical Model.

6.4.1 Indexing

After preliminary lexical processing, Rosie establishes that the linguistic input contains two referring expressions (REs: *the red cylinder* and *the blue triangle*), a spatial preposition (*to the right of*), and a verb (*move*). The goal of the *indexing* step is to identify the referents for these linguistic units. In the following text (and the Algorithm 1), R_{sub}^{super} denotes the referent set. The superscript $super$ denotes the contents of the set (o for objects, s for spatial relations, and a for actions/tasks) and the subscript sub denotes the words used to generate the set.

To index REs (*the red cylinder*, procedure $\textbf{index-RE}$ in Algorithm 1), the model must first index the descriptive words (*red* and *cylinder*). For each of these words, the model queries the semantic memory for a node that was previously learned to be associated with the lexical string. For the string *red*, the memory returns node `L1` (refer to Figure 6.1). Node `L1` *maps* the lexical string *red* to the corresponding perceptual symbol `C22` which is a class in the color classifier. Once the model has retrieved perceptual symbols for all words, it searches working memory for objects that have the required perceptual symbols. These

Algorithm 1 Comprehending a parsed imperative sentence

procedure index-RE(referring expression re)
  ▷ partially described here, details in Section 6.5
  while re contains a descriptive word w that has not been grounded **do**
    $T \leftarrow \phi$
    if $n \leftarrow$ **smem-query**([lex: w]) not a failure **then** add n.sense.class in set T
    else the noun/adjective w is novel, begin learning interactions
  if $(R \leftarrow o \in O | T \subseteq o) \neq \phi$ **then**
    if $|R|$ = 1 **then return** R
    else apply resolution strategy described in Section 6.5
  else described object cannot be recognized, ask for examples
procedure index-preposition(preposition p)
  if $n \leftarrow$ **smem-query**([lex: p]) not a failure **then return** $\{n\}$
  else the preposition p is novel, begin learning interactions
procedure index-verb(verb v)
  $n \leftarrow$ **smem-query**([verb: v])
  $m \leftarrow$ **smem-query**([lex: n])
  return $R \leftarrow \{(m.\text{lex}, m.\text{proc}, m.\text{goal})\}$
procedure index-imperative(sentence s)
  if direct object do of s has not been grounded **then**
    $R^o_{do} \leftarrow$ **index-NP**(do)
  if s contains a preposition phrase pp that has not been grounded **then**
    if preposition p in pp has not been grounded **then**
      $R^s_p \leftarrow$ **index-preposition**(p)
    if object po in pp has not been grounded **then**
      $R^o_{po} \leftarrow$ **index-NP**(po)
    $R^r_{pp} \leftarrow ([\text{relation: } r^s_p], [\text{object: } r^o_{po}]) | r^s_p \in R^s_p \wedge r^o_{po} \in R^o_{po}$ ▷ assumption: $|R^s_p|$ = 1
  if verb v of s has not been grounded **then**
    $R^a_v \leftarrow$ **index-verb**(v)
  $(lex, proc, goal) \leftarrow r_v \in R^a_v$ ▷ assumption: $|R^a_v|$ = 1
  $slot \leftarrow lex$.direct-obj ▷ process direct object
  $arg1 \leftarrow$ edge of $proc$ that connects to $slot$
  $A_{arg1} \leftarrow [arg1 : r_{do}] | r_{do} \in R^o_{do}$
  if s contains a preposition phrase pp **then**
    $slot_o \leftarrow lex$.pp-obj.obj
    $arg2 \leftarrow$ edge of $proc$ such that $proc.edge$.object connects to $slot_o$
    $A_{arg2} \leftarrow [arg2 : r_{pp}] | r_{pp} \in R^o_{pp}$
    $I_s = [\text{name: } proc.\text{name}] \times A_{arg1} \times A_{arg2}$
  else $I_s = [\text{name: } proc.\text{name}] \times A_{arg1}$
  $T \leftarrow t | available(t, s)$ $X = T \cap I_s$
  if $|X| = 1$ **then execute**$(x \in X)$
  if $|X|$ = 0 **then** the described task is unknown, learn it through instruction
  if $|X| > 1$ **then** several instances of the task are applicable, ask further questions to constrain interpretations

objects are assumed to be the intended referents of the RE. In cases where the RE is under-specific (e.g., *this block*), there may be multiple objects that match, resulting in ambiguity. The model can use other kinds of information to resolve such ambiguities as we describe in Section 6.5. For the sake of simplicity, in this example we assume that only one object (O12) matches the cue. This object is included in the referent set ($R^{o}_{red,cylinder}$ = O12) for the RE *the red cylinder*. Similarly, $R^{o}_{blue,triangle} = \{\texttt{O32}\}$ for the RE *the blue triangle*. If these sets are empty, there is an impasse and the model cannot progress further. It indicates that Rosie lacks knowledge to generate groundings of the RE, in which case it prompts the instructor for training examples.

Prepositions are indexed in a similar fashion (procedure **index-preposition** in Algorithm 1). For a preposition string (*right*), the model queries the semantic memory for an indexical node that had previously been learned and is associated with it. On the retrieval of the requested node (*P4*), the model creates the referent set $R^{s}_{right} = \{\texttt{P4}\}$. If the set is empty, Rosie asks the instructor to provide an example of *right-of* in the environment.

To index the verb *move*(procedure **index-verb** in Algorithm 1), the model queries the semantic memory for a node that is connected to the string *move*. The memory returns the node L2. Then the model retrieves the mapping node M2 that associates the verb to domain knowledge of the task -- the goal definition G2 and procedural operator node P2. The referent set for the verb consists of the task-concept network, $R^{a}_{move} = \{\texttt{(L2,P2,G2)}\}$.

6.4.2 Instantiating Domain Knowledge

Once the referents have been identified, the next step is to retrieve the domain knowledge associated with them and instantiate it under the syntactical constraints of the sentence (Lines 26 - 35 in Algorithm 1). The model begins by retrieving the previously learned syntactical nodes associated with the verb *move*. The sentence *move the large red cylinder to the right of the blue triangle* has a direct object (RE, *the large red cylinder*) and a prepositional object (RE, *the blue triangle*) connected to the verb through the preposition *right*.

Following this syntactical structure, the model retrieves the direct object (`direct-obj` in the figure) node `A11` and the prepositional phrase object (`pp-object` in the figure) node `L3` which is further expanded to retrieve nodes `A21` and `A31`. The slot nodes `A11` and `A21` can receive sets of objects in the environment. `A11` is filled by $R^{o}_{red,cylinder}$ as *the red cylinder* is the direct object of the verb *put* and `A21` is filled by $R^{o}_{blue,triangle}$, as *the blue cylinder* is the RE in the prepositional phrase of the verb. `A31` is a spatial slot that is filled by R^{s}_{right}, the referent set for *right*.

Next, the model expands the domain knowledge nodes `P2` and `G2`. The subgraph (`P2, P3, A11, A21, A31`) constrains how the policy operator `op_1` is instantiated. The subgraph (`G2, G3, A11, A21, A31`) constrains the instantiation of goal of the task. The values of the slot nodes (`A11`, `A21`, `A31`) determine the contents of the goal and the policy operator `op_1`. Instantiation of domain knowledge results in the interpretation set I_s. This set contains the policy `op_1` defined over objects drawn from sets $R^{o}_{red,cylinder}$, R^{s}_{right}, and $R^{o}_{blue,triangle}$ which is executed until the corresponding goal is achieved.

In Glenberg and Robertson's (1999) formulation of the Indexical Hypothesis, this step was described as *deriving the affordances*. However, the term *instantiating domain knowledge* better describes our formulation of the process.

6.4.3 Meshing

The interpretation set I_s is the set of different groundings of the imperative sentence, which can have several elements arising from under-constraining cues in the linguistic input. However, only a subset of these groundings can be executed in the environment given the physical constraints and spatial relationships between objects. For example, the *open* action can only be executed for *stove* and *pantry*. When *open* is used with an under-constraining RE such as *it*, there will multiple interpretations, but only two of those interpretations can be executed in the environment. This implements the state-sensitive selectional restrictions of verbs described in Chapter 5.

The meshing step is described in Lines 36 - 39 in Algorithm 1. Suppose T is a set of tasks that can be executed in the current state based on their availability conditions. The intersection set $I_s \cap T$ is the set of tasks that the instructor intends Rosie to execute. If this set contains a single element, that task operator is selected and executed. If this set contains multiple elements, further interaction or internal reasoning is necessary for resolution. The cardinality of the referent sets (R) is used to determine the source of the ambiguity. Rosie asks questions to gather information that will reduce the cardinality of the ambiguous set. If the $I_s \cap T = \phi$, Rosie does not have enough knowledge to generate the correct groundings for the required task. This is an opportunity to learn the task, so Rosie begins a learning interaction by prompting the human collaborator to present an example execution.

6.4.4 Active and Expandable

The Indexical Model proposed here formulates the problem of comprehension as searching the common ground for referents using words as cues and then composing them under the constraints imposed by the syntax and what is afforded in the environment. The search for a referent is a failure if Rosie lacks knowledge of how to associate words with non-linguistic knowledge. For example if it does not know which perceptual symbol the word *red* maps to, it fails while trying to index it and an impasse occurs. Soar's state stack contains information about the processing phase during which the failure occurred. This information is not only useful in constructing an informative query (*What is red?*) but also is useful in learning from instructor's response. The newly acquired knowledge then can be immediately exploited for making progress on comprehension. Consequently, the Indexical Model expands to incorporate newly acquired knowledge (D5) without ever requiring Rosie to go offline to learn grounding knowledge. This is in contrast with related work (D. Chen and Mooney, 2011; Tellex et al., 2011) on grounded comprehension that fails while comprehending words that the system wasn't trained on. The Indexical Model

provides information that is not only useful in progressing the interactive dialog forward but also in integrating incremental instructions in a comprehensive task-oriented representation (D4).

We constructed a corpus of task commands by composing nine nouns and adjectives, three prepositions, and three verbs. Rosie was asked to execute the task commands one-by-one. It began with no prior knowledge of any of the words. In order to correctly comprehend the task command and execute the task, Rosie must learn how to ground words into its perceptual, spatial, and task knowledge.

Figure 6.2 shows the Rosie's performance in comprehending and executing tasks. The data is representative of Rosie performance over several runs. To execute the first command, it initiates 24 interactions as it fails to ground the noun (*rectangle*), adjectives (*blue*, *small*), and verb (*move*) and attempts to learn them. The knowledge acquired is general and the Indexical Model can use this knowledge for interpretation of other task commands. For example, the spatial and task knowledge learned for *left-of* and *move* is useful in comprehension and execution of task command 15. The interactions for task command 15 pertain only to learning new adjectives *large* and *red* and the new noun *arch* and therefore, are less than those required to execute task command 1. At task command 25, Rosie has learned all elements of the dataset and does not require any learning interactions.

Note that the reduction in number of interactions as the experiment progresses is a consequence of the dataset used. If after task command 25, new verbs, adjectives, nouns, or prepositions are introduced in the dataset, the number of interactions will increase. This is because the Indexical Model will fail at grounding the novel words and Rosie will begin interactions to learn their groundings. However, after learning this knowledge, the Indexical Model will be able to use this knowledge for comprehension of other task commands that use these novel words. This occurs online.

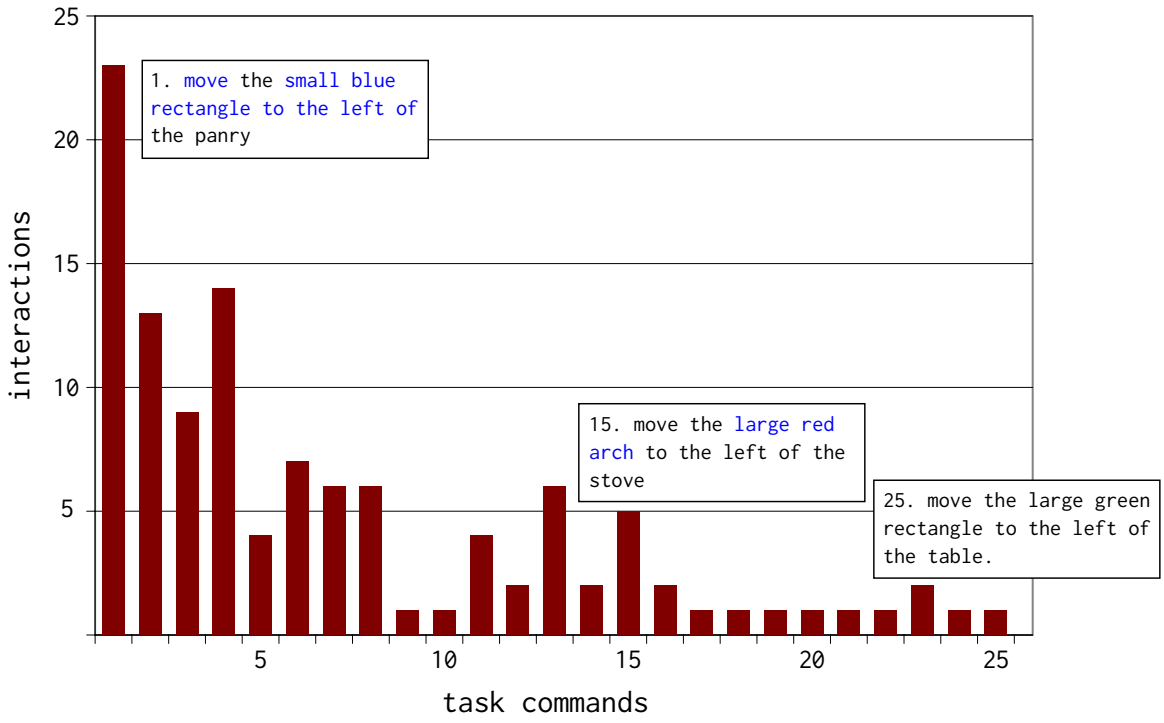

Figure 6.2: Number of agent-initiated interactions per task command. The boxes show task commands 1, 15, and 25. The words in color blue are unknown to Rosie when that task command was given.

6.5 Reference Resolution

Humans use a variety of surface forms to refer to the same entity. A few, such as definite noun phrases (*the large red cylinder on the table*), may uniquely identify the intended referent from the current shared perceptions. However, the majority of referring expressions (REs) encountered in conversations, such as noun phrases with indefinite determiners (*a cylinder*), demonstrative/diectic pronouns (*this*, *that*), and personal pronouns (*it*), are ambiguous.

For the generation and comprehension of REs, the communicative goal is the identification of the intended object by the hearer. The form of REs and other linguistic (word order) and phonetic (intonation) aspects are influenced by the cooperative speaker's assumptions about the relative salience of referents to the hearer. An object might be more

salient than others because it is useful in performing a task, it is being pointed at, it changes appearance, or it is unexpected. The ongoing discourse can also make objects more salient. Speakers make assumptions about which objects are more salient to the hearers and use these assumptions to choose an appropriate RE. More salient objects can be referred to by less informative REs, as the hearer can exploit saliency for disambiguation. This leads to efficient (fewer symbols) communication.

Gundel et al. (1993) express the notion of the current and historical salience of an object to the hearer as its *cognitive status*. They propose a Givenness Hierarchy (GH) that relates the cognitive status of objects with different RE surface forms. The GH identifies six cognitive statuses, only four of which are relevant to our domain:

> *in-focus* (personal pronouns) > *activated* (demonstrative pronouns, demonstrative noun phrases) > *uniquely-identifiable* (definite noun phrase) > *type-identifiable* (indefinite noun phrase).

Each status in the GH is the necessary and sufficient condition for use of the corresponding RE and entails all the lower statuses. The choice of a RE form by the speaker is indicative of which cognitive status is useful for resolution. Given the cognitive status of an object and the hearer's knowledge about the environment, the information in the RE uniquely identifies the intended referent.

6.5.1 Non-linguistic Contexts

Knoeferle and Crocker (2006) identify two dimensions of the interaction between the linguistic and situated context: *informational* and *temporal*. The first dimension refers to the rapid integration of diverse information for various cognitive modules including perceptual processes and domain knowledge. The second dimension refers to the temporal coordination between attentional processes and utterance comprehension. While REs such as noun phrases (lower in the GH) exploit the informational dimension of language-context interaction, ambiguous REs such as pronouns (higher in the GH) exploit the temporal

dimension. To process the complete range of RE forms in the GH, the Indexical Model exploits both the informational (described previously) and the temporal dimensions (described below).

- *Interaction*: When conversation participants communicate, they focus their attention on only a small portion of what each of them perceives, knows, and believes. Some entities (objects, relationships, actions) are central to information transfer at a certain point in dialog and, hence, are more salient than others. This is exploited by both the speaker and the hearer. It lets the speaker refer to focused entities with minimal information and lets the hearer heuristically constrain the set of possible referents, reducing cognitive load on both.

  Rosie has a model of instructional interaction that is based on a collaborative discourse theory by Grosz and Sidner (1986). The details of this model are in Chapter 7. The model organizes the discourse structure according to the goals of the task and maintains a set of all referents (objects, spatial predicates, actions) that are related to the ongoing dialog. The set of objects (O^{stack}) is most pertinent to this chapter because it identifies all the objects that have been referred to in the current discourse, making them more salient than other perceivable objects.

- *Attention*: Object referents that have been brought to attention, either through linguistic or extra-linguistic means, but are not in the focus of the ongoing communication are usually referred to by demonstrative pronouns or demonstrative noun phrases (*this*, *that cylinder*) (Gundel et al., 1993). The extra-linguistic means may include pointing by the speaker or unexpected stimulus such as a loud noise. To resolve such REs, Rosie must maintain the history of references to objects in its perceptions.

  Rosie uses the architectural recency-based activation in Soar's semantic memory as a form of attention. The recency-based activation biases retrieval from semantic

memory towards the more recently accessed elements. An object is *accessed* only if it was pointed at or was used in an action or learning. Anytime an object is accessed, Rosie stores its representation in the semantic memory which boosts its activation in accordance with recency computation. A completely ordered subset O^{active} of the highest activated n objects is retrieved from Rosie's semantic memory to its working memory. These are combined with objects in focus to give a set of objects to which Rosie is *attending* ($O^{attend} = O^{stack} \cup O^{active}$). This formulation of attention combines linguistic and extra-linguistic notions of salience.

6.5.2 Resolving References in the Indexical Model

In Section 6.4.1, we described the indexing of referring expressions in simple cases where the words in the RE and their corresponding perceptual symbols by themselves uniquely identified the referent object. Here, we give details about how an ambiguous RE is indexed by incrementally adding diverse types of information in the Indexical Model. The steps below implement the **index-RE** procedure in Algorithm 1.

1. *Maintain cognitive status.* Following the Givenness Hierarchy, the model maintains different cognitive statuses for objects:
    - Objects in the interaction stack (O^{stack}) have the *in-focus* status;
    - Objects that are being attended to (O^{active}) have the *activated* status;
    - Objects in perceptions ($O^{percept}$) have the *identifiable* status.
2. *Assign resolution type.* For any RE r, the model determines its resolution type based on its surface form. If the RE is:
    - a definite noun phrase (*the red cylinder*), demonstrative pronoun (*this*), or personal pronouns (*it*), the speaker has a specific intended referent and comprehension should unambiguously determine it (*unique* resolution);

- an indefinite noun phrase (*a red cylinder*), this indicates that there is no specific intended referent and any object that fits the noun phrase can be used for resolution (*any* resolution).

3. *Determine the candidate referent set.* The model uses the RE surface forms to determine which set contains the intended referent. The candidate referent set is:

   - $R_r^o = O^{stack}$ for personal pronouns (*it*);
   - $R_r^o = O^{attend}$ for demonstrative pronouns (*this*, *that*) and noun phrases (*this cylinder*);
   - $R_r^o = O^{percept}$ for definite (*the cylinder*) and indefinite (*a cylinder*) noun phrases.

4. *Apply the visual filter.* ROSIE's knowledge of the perceptual symbols and how they relate to words is useful in identifying the referents of descriptive REs (*the red cylinder*). The model indexes each descriptive word (*red*, *cylinder*) in a noun phrase, and then the model looks up its corresponding perceptual symbols, which are collected into a set as a cue. All the objects in the candidate set (R_r^o) whose working memory representations do not contain this cue are deleted from this set.

5. *Apply the spatial filter.* If the RE uses spatial reference (*the cylinder on the right of the pantry*), referent sets for both noun phrases ($R^o_{cylinder}$, R^o_{pantry}) are obtained. The model indexes the preposition *right* to retrieve the corresponding spatial relationship predicate `P4`. Items in $R^o_{cylinder}$ that do not satisfy the relationship `P3` with any item in R^o_{pantry} are deleted. This is a meshing step that combines linguistic information with the domain knowledge and the perceptual state.

6. *Apply the task filter.* If the REs are used with verbs, as in an action command (*put the cylinder in the pantry*), the model uses the knowledge of task restrictions to constrain their interpretation. To access this knowledge, the model indexes the verb to retrieve a task-operator and its corresponding goal. During meshing, it looks at all

task-operator instantiations that are applicable in the current environmental state under the physical constraints and the knowledge of object affordances. Any object that does not occur in the arguments of currently applicable task instantiations is removed from R_r^o of the RE.

7. *Obtain partial ordering.* The elements of the referent set ($r \in R_r^o$) are partially ordered based on their cognitive status and resolution type. If resolution is *unique* (from step 1), then $r_i \in O^{stack} > r_j \in O^{active} > r_k \in O^{percept}$. If resolution is *any*, then all objects have equal preference.

8. *Resolve.* After applying all available filters, if R_r^o contains only a single object, that object is selected as the intended referent. If it contains multiple objects, the model uses the partial ordering obtained earlier to select the object highest in the order as the intended referent. If the partial ordering is not informative enough for resolution, the model initiates a subdialog to obtain more information from the instructor. If the resolution is *any*, all objects have equal preference and one is chosen at random.

The resolution process described here integrates seamlessly with the incremental learning modules and the interaction model for mixed-initiative conversations.

6.5.3 Integrative Processing

Human situated communication exploits non-linguistic contexts to convey meaning. Often, the speaker omits information that is apparent from perceptions, common knowledge, or shared experience and relies on the hearer to infer it. This results in efficient (fewer words) but linguistically ambiguous communication. The use of under-specific referring expressions such as *it* or *this cylinder* that by themselves do not uniquely identify the referent is an example of this. An intelligent agent that effectively engages in linguistic communication with a human must be able to handle such omissions undertaken for

efficiency.

The RE resolution in the Indexical Model proposed in this chapter can exploit a variety of non-linguistic contexts (informational and temporal) to address object reference ambiguity in natural language. We demonstrate this through the experiment below where we construct scenarios with different levels of ambiguity in object reference and record the behavior of different variation of Rosie that use varying amounts of non-linguistic contexts.

6.5.3.1 Experiment

Dataset: We generated a corpus of 25 instructor utterances that addresses different capabilities of the agent. This corpus contains instruction sequences that teach and query Rosie about objects and their attributes, present and verify grounded examples of spatial prepositions, and teach verbs. This corpus contains references to three objects on the scene. These objects are referred to using varying forms of referring expressions including 12 instances of personal pronouns (such as *it*), 4 instances of demonstrative pronouns (such as *this*), 3 instances of demonstrative phrases (such as *that cylinder*), and 14 varying length noun phrases with different descriptive words (such as *the red cylinder*). Note that this corpus consists of utterances by the instructor only. As these instructions are provided to the agent, it engages the instructor in subdialogs for correctly identifying the referent and for learning verbs. The length of these dialogs vary with different models of comprehension and scenarios (described below). From the 16 dialogs (4 models $\times$ 4 scenarios), the longest dialog had 178 human/agent utterances.

Models: We evaluated various models of comprehension that exploit different dimensions of language-context interaction. The baseline model *p* uses the context derived from perceptual semantics only. To obtain other models, we incrementally added other kinds of contexts. Model *p+t* exploits the restrictions derived from task knowledge along with perceptual semantics. Model *p+t+a* exploits the temporal dimension by encoding the at-

tentional state. Model *p+t+a+d* encodes both the attentional and dialog states.

Scenario Ambiguity: Each of the comprehension models was evaluated using the instruction corpus on different scenarios of increasing perceptual ambiguity in the environment obtained by adding distractor objects. The *ambiguity 1* scenario only contained the intended referent objects on the scene. *Ambiguity 2* contained distractor objects that were perceptually distinct (different colors, shapes) from the intended referents. *Ambiguity 3* contained distractor objects that were of the same color as the intended referent but of different shapes. *Ambiguity 4* contained distractor objects that were perceptually similar to the intended referents and required the use of spatial references.

Evaluation metric: We seek to measure the length (number of lexical symbols) of the *optimal* RE for a model in a scenario - the shortest RE that uniquely identifies an object without any further interactions about it. Other aspects of communications being the same, the model that has shorter optimal REs can sustain more efficient communication. We arrive at this measure indirectly as described below.

Rosie is an interactive agent that engages the human instructor in a subdialog if it fails at any stage in its processing. On failing to resolve ambiguous referring expressions in sentences, Rosie asks questions to obtain more information that will constrain its resolution. The instructor can, then, incrementally provide more identifying information. An example dialog is below.

Instructor: *Pick it up.*
Agent: *Which object?*
Instructor: *the blue one.*
Agent: *Which blue object?*
Instructor: *the cylinder.*
Agent: *Which blue cylinder?*
Instructor: *the one in the pantry.*

The question-answer pairs (*object identification* queries) are informative of how ambiguous an RE is given the ambiguity in the current scenario and the contexts. The instructor could have provided all the identifying information in a single response (*Which object?*,

the blue cylinder in the pantry). However, letting Rosie take the initiative in resolution ensures that it accumulates the minimum information required for unique identification in the current situation. The number of *object identification* queries in this setup correlates with the length of the *optimal* RE for the agent in the particular scenario.

6.5.3.2 Results

The graph in Figure 6.3 shows the number of *object identification* queries asked by Rosie while using different comprehension models in scenarios with varying perceptual ambiguity. The models reliably integrate information provided incrementally over several interactions for resolution. Consequently, all REs were eventually correctly resolved in all models in all scenarios. The model *p+t+a+d* can exploit the informational and temporal dimensions effectively for resolution.

The baseline model *p*, which only exploits the contexts derived from perceptual semantics, generates the most queries for all levels of ambiguity. Model *p+t* is able to use its knowledge about the task to constrain resolution and, therefore, requires fewer queries (has shorter optimal REs) for achieving the same resolution results. The models that exploit both the temporal and informational dimensions require even fewer queries to achieve similar performance across all scenarios. Conversing with agents that only encode the informational dimension of non-linguistic context usually requires wordy REs, such as *the red cylinder in the pantry*, that must be repeated in all interactions related to that object. The use of the temporal dimension for comprehension allows the use of shorter referring expressions (*it*, *this cylinder*), resulting in efficient communication.

As perceptual ambiguity in the environment increases, models that exploit only the informational dimension (*p*, *p+t*) require more perceptual information for resolving REs. Models that exploit the temporal dimension (*p+t+a*, *p+t+a+d*) ask the same number of queries across all scenarios, demonstrating that the use of co-reference is an efficient (uses fewer lexical symbols) way to communicate about objects in human-agent dialogs. It lets

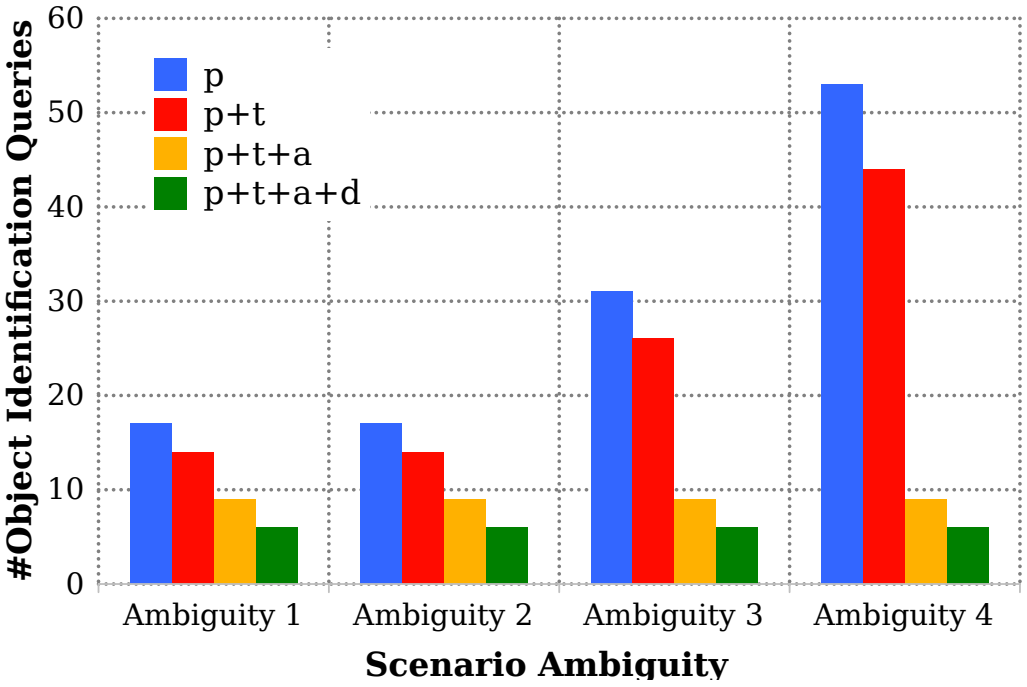

Figure 6.3: Number of object queries asked by Rosie for RE resolution.

the instructor communicate the intended referent without incorporating large amounts of information in utterances in perceptually ambiguous scenarios.

To establish that the non-linguistic context contributes information above and beyond what is encoded in the linguistic features, we ran Stanford CoreNLP (Lee et al., 2012) on our corpus. Co-reference resolution in CoreNLP incorrectly resolved ten (28.6%) references. Features used in CoreNLP for reference resolution roughly correspond to the interaction context used in our model and are derived from several heuristics about coreference. The majority of the errors in CoreNLP are pronoun errors and arise due to the way in which it evaluates potential antecedents (previous reference to the same object). CoreNLP biases resolution of *it* to an antecedent closer in history. This bias may result in incorrect resolutions. Consider the teaching interaction below.

Instructor (points to `object1`): *This is red.*
Rosie: OK.
Instructor (points to `object2`): *This is blue.*

CoreNLP resolves the first *this* to a new object, which can be correctly resolved to `object1` after some additional reasoning. It, then, incorrectly resolves the second instance of *this* to the antecedent (first *this*) in the first instruction and consequently to `object1`. This occurs because CoreNLP processes a pronoun by collecting references made to objects previously and picking one based on its heuristics. As the heuristics only capture linguistic features, the fact that the instructor pointed to an object is not incorporated in the resolution.

In the Indexical Model, the activation of objects changes as they are pointed at, acted upon, or are used in learning. This activation plays a role in resolution (attentional context). In the example above when the instructor points to object2, it becomes highly activated and the most promising candidate. Similarly, task knowledge of which objects can be picked up plays are role in resolving pronouns. CoreNLP on the other hand relies on the order antecedents are presented resulting in resolution to objects that cannot be picked up.

The results presented here are expected: the model with more knowledge and reasoning capabilities can sustain more efficient communication. However, prior work on situated comprehension has largely ignored the role of non-linguistic knowledge and problem solving in language understanding. The Indexical Model proposed here facilitates the incorporation of reasoning and problem solving in comprehension. The results presented here show that these contexts can be useful in handling ambiguities. Finally, the non-linguistic contexts presented here are only a subset of what is required to sustain a human-like conversation. Other kinds of reasoning and inference may also find use in comprehension. Future work will investigate what reasoning is useful for comprehension and how it can be incorporated in the Indexical Model.

6.6 Unexpressed Argument Alternations of Verbs

In Rosie, the goal of comprehension of an imperative sentence is to correctly instantiate a task that can be executed in the environment. The verb of the sentence identifies the task and the verb's objects identify the arguments of the task. The syntax is useful in instantiating the task goals and a policy that can be executed in the environment to achieve them. However, as explained in Chapter 5, several verbs have unexpressed object alternations. A example is the verb *take* which is used in two alternations:

1. *Take the trash out to the curb.*
2. *Take the trash out.*

The second alternation leaves the location unexpressed. Humans generate and comprehend such sentences by relying on the shared knowledge about the domain. In the example, both the speaker and the hearer know that the *trash* is usually put on the *curb*. This lets the speaker omit the location in the sentence *take the trash out* for the sake of communicative efficiency. The choice of this syntax by the speaker indicates that they assume the hearer can fill the missing location from their knowledge of the domain. Upon hearing the utterance, the hearer must exploit this knowledge and generate an appropriate, complete representation of the task.

6.6.1 Exploiting the Hearer's Instructional Experience

To deal with imperative sentences with unexpressed information about the action, the model relies on Rosie's task representations learned through interactions with the instructor described later in Chapter 8. Here we briefly summarize the process. Consider the verb *move* and the variations of imperatives that can be constructed from it:

(a) *Move the green object to the right of the table.*

(b) *Move the green object to the table.*

In (a), the direct-object *the green object*, the location *the table*, and their spatial relationship (*right of*) are completely specified. In (b), the spatial relationship is omitted with an understanding that there is a default configuration (*on*) between the object and the location.

The default configurations are learned from multiple instructional task executions of the task. The default configuration can be extracted from the experience of learning how to perform the *move* task. When Rosie is asked to execute a task for the first time, it leads the instructor through a series of interactions to learn the structure of the task. Suppose that Rosie does not know how to perform *move*. On receiving the imperative sentence (a), it asks a question about the goal (*what is the goal of the task?*) and the human instructor replies, *the goal is the green object to the right of the table*. By analyzing the sentence and the goal description, Rosie extracts a general schema that relates the linguistic structure of the utterance to the goal of the task. It uses a simple heuristic that information (object, location, and spatial relationship) specified in the imperative sentence can be generalized away in the goal definition. Rosie assumes that future instances of the verb *move* will completely specify the goal. At a later stage, Rosie receives the sentence (b). Using its knowledge of the goal definition, Rosie attempts to generate an instantiation. This fails because no relationship is specified. So, Rosie asks the instructor to describe the goal. The instructor may reply with *the goal is the green object is on the table*. By comparing the current situation (for sentence (b)) and its experience with sentence (a), Rosie concludes that the verb *move* may be used in two alternations. The representation of *move* is augmented to reflect that, if the relationship is not specified, it should attempt to establish the *on* relationship between the object and the location.

When comprehending the verb *move* in the future, the model can use the default values to complete the argumentation of the action if those values are not specified in the linguistic input itself. These values are available through the goal elaboration operator described in Chapter 5. This lets the model use Rosie's instructional experience to fill in

information that is not specified in the linguistic input but is essential for action.

6.6.2 Integrative Processing

In situations where a task is incompletely specified linguistically, the Indexical Model can employ prior experience with task execution to extract relevant information. We demonstrate this in the experiment below where we report the performance of the Indexical Model on unexpressed verb alternations.

6.6.2.1 Experiment

In an environment with four objects, we instructed Rosie to perform eight instances of five tasks using a uniform distribution over alternations of the relevant verb. Here we characterize the verbs used in the experiments.

- The verb *pick* takes a direct object and does not have any alternation. Example: *pick up the red cylinder.*
- The verb *put* takes a direct object and a prepositional object and does not have any alternation. Example *put down the red cylinder on the table.*
- The verb *move* has two alternations. The first specifies the object, the prepositional object, and the intended spatial relationship (as in *move the red cylinder to the right of the table*). The second does not specify the spatial relationship between the direct and prepositional object (as in *move the red object to the table*).
- The verb *store* has two alternations. The first specifies the direct object, the prepositional object, and the intended spatial relationship between them (as in *store the red cylinder in the pantry*). The second leaves the prepositional object unexpressed (as in *store the red cylinder*).

- The verb *cook* has two alternations. One specifies the instrument used for cooking along with the object to be cooked (as in *cook the steak on the stove*). The other leaves the instrument unexpressed (as in *cook the steak*).

The first two verbs are primitives that have been pre-encoded in Rosie; the last three are acquired through human-agent linguistic interaction. For training, Rosie was taught the task with the first alternation of the corresponding verb. After it successfully learns the task, we asked it to perform the task using the second alternation. Any questions asked by Rosie during this training episode were appropriately answered. Two variations of the comprehension model were evaluated. Model+e uses Rosie's instructional experience to augment the linguistic input that is missing information required for task execution. Model-e is a lesioned version of model+e that does not exploit the instructional experience but relies on asking the instructor a question for the missing information. Both models were given the same instructional experience (12 interactions for *move* and 16 interactions for *cook*).

6.6.2.2 Results

The graph in Figure 6.4 shows the number of interactions that occurred during the comprehension of task commands in model+e (in blue) and model-e (in red). The patterned bars correspond to the first alternation and the plain bars correspond to the second alternation (if applicable). For verbs without alternations (*pick* and *put*), both models take equal number of interactions to execute the task (one per task instance). For verbs with alternations, the models behave differently for different alternations. For the first alternation in which all information is specified, both models take one interaction per task. However, for the second alternation that leaves some argumentation unexpressed, model+e takes only one interaction per task because it uses the knowledge acquired through learning to fill in the missing information. Model-e must ask questions to gather the information missing from the sentences with unexpressed verb argumentation, resulting in more human-agent in-

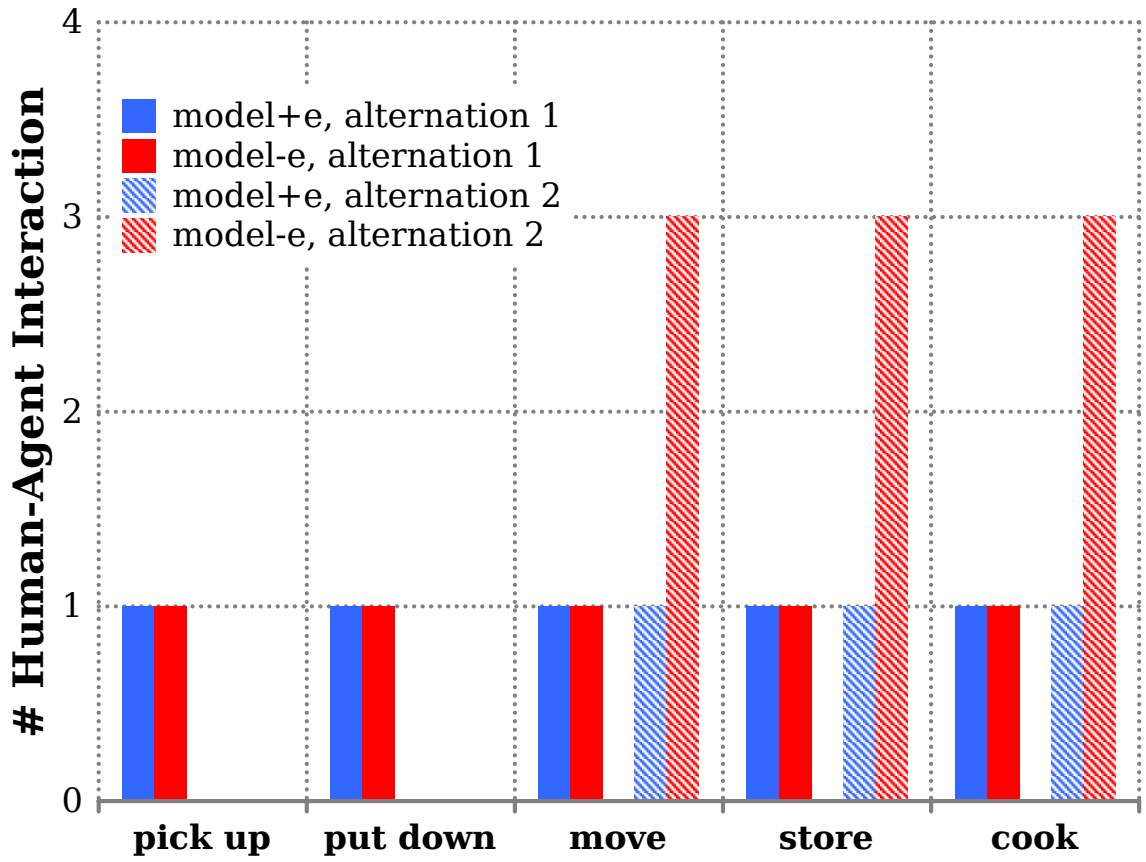

Figure 6.4: Number of interactions required for comprehending verbs in different alternations.

teractions (three per task instance). Both models comprehend both alternations of verbs and correctly execute the task.

The method and analysis presented here are fairly simplistic. This is a product of both the domain and our limited understanding of when and why information is omitted in communication. Although, prior work in linguistics (Levin, 1993) presents a detailed analysis of what kinds of verb objects may be left unexpressed, it does not discuss why this occurs. The issue of unexpressed verb objects and its impact on grounded verb representation and processing has not been addressed by prior computational work on situated comprehension.

6.7 Summary and Discussion

The Indexical Model proposed in this chapter implements referential comprehension by formulating comprehension as search over perceptions, short-term memory, and long-term knowledge (D2). In this chapter, we introduced *indexical maps*, structures in Rosie's long-term semantic memory that align linguistic symbols and constructions to those that represent modal knowledge about the world. Using a simple referential grammar, the model can search its semantic memory using words such as *red* and can access a perceptual symbol `C22` that refers to a class in Rosie's perceptual memory. `C22` can be used to search the working memory for all objects that are *red* in order to comprehend *pick up the red block*.

In formulating comprehension as a search over short-term and long-term experiential knowledge, non-linguistic context has a natural role (D3). It provides constraints over the hypothesis space and guides search. Non-linguistic context can be derived from various sources including the ongoing discourse, the current perceptual state, the knowledge of tasks, and the models of environmental dynamics. Other cognitive mechanisms such as reasoning and attention also contribute to comprehension by providing additional constraints on interpretations. We have shown that exploiting different contexts in the Indexical Model reduces ambiguity in referring expression resolution. Experiential knowledge augments the linguistic input by incorporating knowledge from prior experiences with the environment. This is useful in situations where the linguistic input, such as *take the trash out*, is under-specific and does not encode enough information for reasoning and action.

In comparison to standard approaches to semantics and meaning representations prevalent in natural language community, the Indexical approach to language comprehension affords several advantages. Previous approaches either encode semantics as amodal symbols that are not grounded in real-world experiences or as propositions that do not capture the relational or distributional properties of complex environments. In the Indexical ap-

proach, semantics can be encoded using diverse, modality-specific representations. These include probabilistic representations for perceptions, relational representations for spatial reasoning, hierarchical policies for task execution, and models for reasoning about the environmental dynamics. Such representations are typical of agents designed to function in complex environments. Additionally, established learning algorithms (kNN, version-space learning, explanation-based learning) can be used to expand the agent's knowledge, thereby, extending its situated comprehension capabilities (D5).

The model is implemented in Soar and makes extensive use of meta-cognitive information such as the state stack or the kind of impasse to reason about failures in processing an utterance. This information is useful in determining the cause of failure and in generating appropriate questions in case of ambiguity or missing information (D4).

The focus of our future work will be on studying other linguistic ambiguities that arise in instructional interactions and how they can be addressed by incorporating information from different cognitive modules. Ambiguity may arise in determining the site of prepositional phrase attachment. In the sentence *store the red cylinder on the green block in the pantry*, it is unclear if the phrase *in the pantry* attaches to the verb *store* directly, or to the phrase *on the green block*. This can be resolved by the incorporating the current state of the environment. Another concern is that the proposed model does not support interpretation of quantification (*store all red objects*) or of categories of objects (*chess pawns cannot move backwards*). This limits what can be expressed in instructions for tasks requiring them to be specific to perceivable objects instead of describing general characteristics and rules. Another direction for future research is incremental comprehension (D6) which can provide useful constraints on linguistic perception. This will lead to robustness to noise in speech and better performance on incomplete and ungrammatical linguistic input.

This chapter has focused on mechanisms that are useful in language comprehension where the elements of an utterance can be directly grounded in the shared state between the speaker and the hearer. However, much of human communication is non-situated as

when people talk about scenarios that are not directly perceived and have occurred in the past (retrospective) or may occur in the future (prospective). In such cases, human hearers readily generate perceptual simulations guided by the content of the utterance and use these models to reason about the scenario being described. The perceptual simulations are informed by the hearer's experience of the world. In the future, we will expand the indexical approach to address non-situated comprehension and its interaction with knowledge acquisition and learning.

Chapter 7
Maintaining Flexible Dialog

Situated interactive instruction is a powerful learning paradigm. It distributes the onus of learning between the instructor and the learner. In the SII approach, an instructor can be freed from using a specific ordering to teach the agent new words and concepts. Often in human-controlled interactive learning, the instructor must attempt to build and maintain an internal model of what the agent knows and doesn't know to give it good examples. This is especially challenging when the agent is dynamically learning a variety of concepts (perceptual, spatial, task knowledge in Rosie) from real-world data. In contrast, with mixed initiative interaction, the instructor can rely on the agent to initiate an interaction when needed. This approach can speed instruction by eliminating the need for the instructor to carefully structure the interaction or repeatedly check with the agent to ensure it has completely learned a concept. The agent can actively seek examples of concepts that are hard to learn and avoid asking for multiple examples of easily acquired concepts. The instructor can take initiative in presenting interesting examples to the agent that it might have overlooked, refining agent's learning.

To support SII so that the onus of learning is distributed between both participants, a critical capability is maintaining a flexible task-oriented dialog with the human instructor. This capability provides temporal context for situated comprehension along with contributing to accumulation of common ground (as in Chapter 6) and allows Rosie to ask questions while learning tasks (as in Chapter 8) or other aspects of its world (Mohan, Mininger, Kirk, et al., 2012; Kirk and J. Laird, 2014). It also allows the instructor to the

structure the task instruction in different ways.

This chapter describes how Rosie maintains a mixed-initiative interaction with the instructor and how the interaction model is integrated with comprehension and learning. In the following sections, we give an overview of Grosz and Sidner's (1986) theory of collaborative discourse that stresses the role of intentions (Section 7.1), describe Rosie's interaction model (Mohan, Mininger, Kirk, et al., 2012; Mohan, Kirk, et al., 2013) in Section 7.2, present empirical data (Section 7.4), and discuss the degree to which the interaction model satisfies the SII desiderata (Section 7.5).

7.1 Collaborative Discourse Theory

Grosz and Sidner (1986) posit that all discourse in some sense is *task-oriented* - discourse participants communicate with each other in pursuit of tasks which may pertain to manipulating and navigating their environments, changing each other's belief states, or comprehending language. The main thesis of the work is that the structure of any discourse can be considered as a composition of the following three related elements.

- *Linguistic structure.* This refers to the linguistic aspects of the discourse and captures how the sequence of utterances in the discourse are organized. Utterances in a discourse can be aggregated into *discourse segments* that fulfill a certain function with respect to the overall discourse.

- *Intentional structure.* A participant that initiates a segment does so with a *purpose* (or intention). The purpose provides a reason why this particular information is being conveyed as opposed to some other information and why a communicative action is being performed as opposed to some other action. The purpose specifies how the ongoing segment contributes to achieving the overall purpose of the discourse.

- *Attentional state.* This is an abstraction of the participants' focus of attention as the discourse progresses. This contains entities (objects, events etc.) that are salient

because either they were mentioned explicitly or were useful in producing or understanding the utterances in the discourse.

The rest of this chapter describes Rosie's interaction model that is based on the collaborative discourse theory (CDT).

7.2 Our Approach

Prior work on collaborative discourse theory formulates a model of collaborative planning (Grosz and Sidner, 1986; Grosz and Kraus, 1996) for agents and applies the model to develop a collaboration manager that provides intelligent assistance to air travel (Rich and Sidner, 1998). This model assumed a closed knowledge-set - the agents were preprogrammed with all the knowledge they require to act in their environment. In order to design an interactive learning agent, the model of interaction should accommodate a growing knowledge-set. No prior work has investigated if CDT can be used in the design of interactive learners since it was proposed. In the next section, we describe how CDT can be extended to support interactive learning. This thesis makes a contribution to interactive learning systems by demonstrating that with a simple extension, CDT is sufficient for the desiderata for SII. In comparison with Chapters 6 and 8, the contributions of the work described in this chapter are relatively minor.

- **Task-oriented** (D7) and **Integrative** (D8). The central thesis of collaborative discourse theory is that all discourse is task-oriented where the participants pursue communicative goals in order to achieve their extra-linguistic goals. A subset of communicative goals pertain to gathering and presenting information critical for task execution. Our model invokes appropriate learning algorithms when it recognizes such goals in the ongoing dialog, effectively integrating learning with task execution and interaction.

- **Flexible** (D9). We demonstrate that the interaction model allows for flexible control of instruction. Rosie can guide conversation to various aspects of the task unknown to it. The instructor can also assume control and introduce concepts to Rosie before they are required for task execution.

7.3 Mixed-Initiative Dialog Model

Our interaction model is adapted from Rich and Sidner (1998) that implements collaborative discourse theory and extends their framework to accommodate learning from situated instruction. Figure 7.1 is an annotated trace of how these concepts are used by Rosie while learning the verb *store*. Rosie also needs to learn the adjective *orange* and preposition *in* in order to learn the verb. The figure shows various elements of how the interaction state is represented. The events column annotates the dialog, actions, and acquisition events, the dialog-class column contains classification of human-agent utterances, the segments column contains contiguous events that are related by a common purpose, and the stack column captures the state of interaction at the specified point.

7.3.1 Interaction State Representation

The model represents the state of ongoing interactions using the elements below.

7.3.1.1 Events

An event causes change, either in the environment (*action-event*), the human-agent discourse (*dialog-event*), or the agent's knowledge (*learning-event*). Action-events correspond to primitive actions or tasks that Rosie can undertake in the environment. Dialog-events refer to utterances in the human-agent dialog. They are categorized into different classes based on their lexical and syntactic structures, such as `get-next-task`, `verb-command`, and `attribute-query` as shown in the left most column of Figure 7.1. A

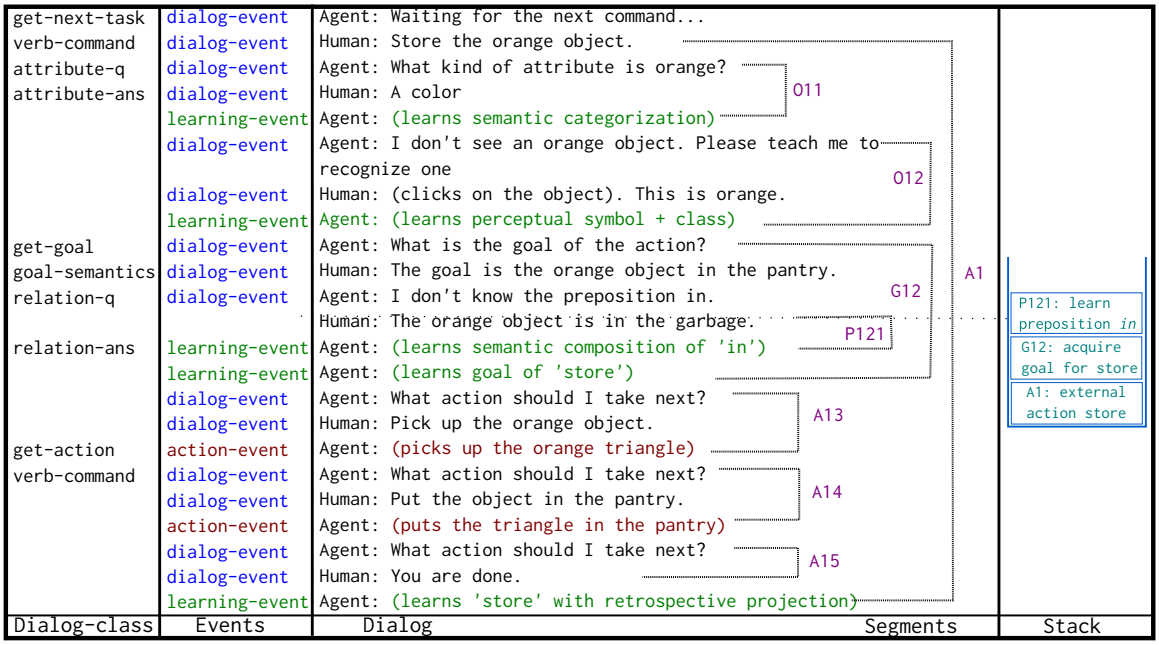

Figure 7.1: Annotated human-agent dialog for acquisition of *store*

.

learning-event is the successful acquisition of perceptual, spatial, semantic, linguistic, or procedural knowledge.

7.3.1.2 Segment

A discourse segment (O12, G12 in Figure 7.1) is a contiguous sequence of events that serves a specific *purpose* and organizes a dialog into purpose-oriented blocks. For example, a question and an answer sequence constitute a segment whose purpose is to achieve shared knowledge between participants; an action-command and an action sequence constitute a segment whose purpose is to change the state of the environment. Segments are hierarchical; a segment is said to be contributing to a parent segment if its purpose contributes to its parents purpose (such as P121 contributing to G12). The segments provide the context for Rosie to organize its processing and interactions in pursuit of its task goals. In our implementation, a segment has the following constituents.

1. *Purpose.* The purpose[1] represents the process goal Rosie attempts to pursue. In our model, they can be of three types: generate a response for instructor's questions, perform the requested task, or learn from provided demonstrations.

   When a purpose is assigned to a segment, Rosie selects satisfy-purpose operators for execution. These operators are responsible for collecting information to reply to instructor's questions, setting up the Rosie's internal state for learning, or for executing a task. The purpose of a segment is heuristically determined based on the fact that Rosie functions in an instructional domain. A collaborator may use a sentence such as *the orange object is in the garbage* to achieve a shared understanding of the perceptual state. In Rosie, this is treated as a beginning of a learning segment in addition to using this information to augment the visual state.

2. *Satisfaction.* The set of events that indicate that the purpose of the segment has been achieved. For example, for a task learning segment, successfully inducing the task representation indicates that the purpose has been achieved.

3. *Cause.* The segments optionally also encode the reason why they were initiated. This informs language parsing, comprehension, and learning. The context of the segment `O11` is useful when parsing the noun-phrase fragment (*a color*) while learning the word *orange*.

When initiated by Rosie, the segments allow the agent to learn a new verb (`A1`), acquire a map for a novel word (such as `O11` and `P121` in Figure 7.1), acquire a goal (`G12`), or acquire a task representation that is required to execute a verb (`A13`, `A14`, `A15`).

7.3.1.3 Active Segment Stack

The attentional structure of discourse is captured in a stack of active segments. When a new segment is created, it is pushed onto the stack. The top segment is the focus of the

[1] We use the term *purpose* to refer to Rosie's internal reasoning goals in order to distinguish them from external task goals such as `(in(A1,pantry))`

current interaction, and Rosie acts to achieve its purpose. When the purpose of the top segment is achieved, it is popped from the stack. The right-most column in Figure 7.1 shows a snapshot of the stack. It contains three open segments, P121, G12, and A1, with P121 being the top segment. The segments are hierarchically ordered with each segment contributing towards achieving its parent's purpose (which is lower in the stack). To learn *store*, the agent must acquire a description of the goal and must learn the spatial concept corresponding to the preposition *in*.

The stack also summarizes all the referents of grounded utterances in the ongoing interaction. This includes all objects and relations referred to along with instantiated tasks being pursued. For situated comprehension, this context is useful in generating hypothesis about objects referred to by REs. For task learning, it provides context useful in combining information provided incrementally into a comprehensive task representation and in creating cues for querying episodic memory during retrospective analysis.

7.3.2 Interaction Management

The interaction model (described in Algorithm 2) changes the interaction state when either the instructor makes an utterance (dialog-event, Lines 6-13 in in Algorithm 2) or Rosie compiles a status while pursuing a purpose (Line 14-18, in Algorithm 2). The status is compiled during the comprehension phase or the behavior phase and contains information about whether or not Rosie was able to successfully complete the phase (Lines 19-27). The success status indicates that either Rosie was able to compile a response to instructor's questions (dialog-event), was able to complete the task (action-event), or was able to learn from the examples given (learning-event). If Rosie fails during any phase, it performs a meta-cognitive analysis in an impasse of why it failed and what information from the instructor will be useful in making progress. A summary of this analysis is included in the failure status.

If the interaction model is triggered, it determines if the event in question is in the

satisfaction set of the topmost active segment on the stack. If it is true, the purpose of the segment is over and the segment is removed from the stack. If it is not true, a rule fires that begins a new segment on the stack and elaborates the purpose of the segment based on heuristics encoded in the model.

Consider the beginning of the interaction in Figure 7.1. When the instructor utters the task command *store the orange object*, the model begins a new segment A1, the purpose of which is to execute the *store* task and its satisfaction set contains an action-event corresponding to the verb *store*. In order to execute the task, a grounded representation of the task command must be created. During situated comprehension, Rosie fails to ground the word *orange*. This results in a failure status that indicates that *orange* is a new noun word. In response to the failure status, the interaction model begins a new segment O1 with the purpose of obtaining the perceptual category of the noun *orange* and with the satisfaction set that includes a dialog-event from the instructor identifying the category. This segment causes a dialog-event by *Rosie - what kind of attribute is orange?*. The instructor responds with another dialog-event - *a color*. The interaction model terminates the segment O1 and simultaneously begins a new segment C11 to incorporate this information in its knowledge about the world. This segment is removed from the stack once acquisition of the perceptual category is successfully completed.

7.4 Evaluation

In the following sections, we demonstrate the interaction model described in the previous sections can sustain a flexible instructional dialog. Further, we show that either participant of the instructional dialog can control the instruction.

Algorithm 2

```
1: procedure insert-new-segment(dialog-event or failure status i, fs)
2:     new ← create-new-segment
3:     new.purpose ← identify-purpose(i, fs.top)
4:     new.satisfaction ← identify-satisfaction-event(i, new)
5:     push(fs, new)
6: procedure interaction-management(instructor's utterance m, focus stack fs)
7:     i ← create-dialog-event
8:     i.message = m
9:     i.category ← classify-interaction(i.message)
10:    if i = fs.top.satisfaction then
11:        pop(fs)
12:    else
13:        insert-new-segment(i, fs)
14: procedure interaction-management(status s, focus stack fs)
15:    if status.event = fs.top.satisfaction then
16:        pop(fs)
17:    else
18:        insert-new-segment(s, fs)
19: procedure satisfy-purpose(stack fs)
20:    proc ← identify-procedure(s.top.purpose)
21:    if event ← execute-procedure(proc) ≠ null then
22:        status ← success
23:        status.event ← event
24:    else
25:        status ← failure
26:        status.context ← metacognitive-analysis
27:    interaction-management(status, fs)
```

7.4.1 Adaptive, Flexible Dialog

The interaction model described in this chapter facilitates a flexible human-agent dialog instead of forcing the dialog to follow any preset script. The dialog contents are influenced by what Rosie knows and does not know about its environment.

7.4.1.1 Experiment

Rosie's prior knowledge was varied to evaluate if the interaction model supports flexible dialog. Prior knowledge states were categorized as:

- `null`: Rosie has no prior domain knowledge beyond the primitive actions
- `O`: Rosie has prior knowledge of how to recognize objects referred to using noun phrases such as *the red large triangle*
- `O+S`: Rosie has prior knowledge of object recognition and spatial relationships such as *in*.
- `O+S+T`: Rosie has structural and execution knowledge of the task in addition to the knowledge of object attributes and spatial relationships.

In these varying initial knowledge states, Rosie was asked to execute three tasks: *place*, *move*, and *store* where *place* was the constituent task for *move* and *store*. The task commands were generated by combining the verbs with various nouns, adjectives, and prepositions. If it asked any questions in executing the tasks, appropriate answers were given.

7.4.1.2 Results

Figure 7.2 shows the number of human-agent utterances required to execute each task in different initial knowledge-states. Human-agent interactions are categorized according

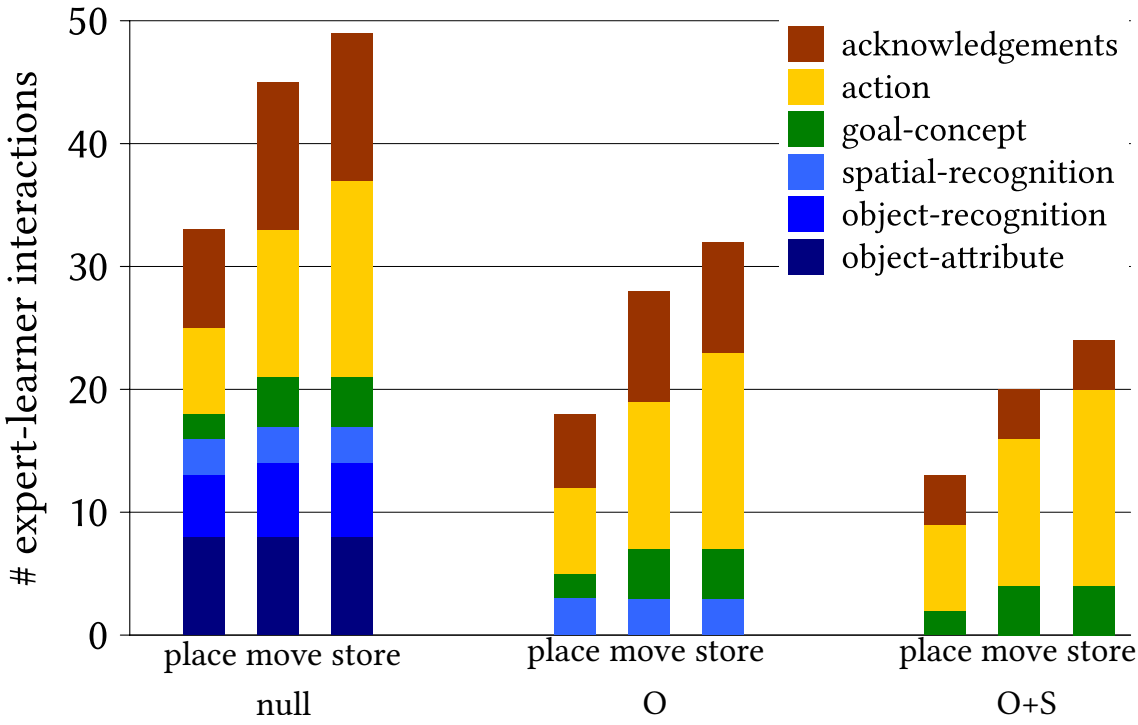

Figure 7.2: Number of agent-initiated interactions per task command in different initial stages of prior knowledge.

to what was the interaction about. For example, the *object-attribute* label refers to interactions that occurred when Rosie learns perceptual concepts such as *red* and how they map to perceptual symbols. When Rosie begins in `null` state, it initiates several discourse segments to learn about objects and spatial relationships. These interactions do not occur if the learner begins in `O+S` states because it already has knowledge of these concepts. In each of these cases, Rosie was able to successfully execute the task. The results show that the human-agent dialog adapts to what is required by Rosie for making progress on the task. The interaction model allows Rosie to change the focus of interaction on various aspects of the task. Further, results show that Rosie only requests knowledge when it is missing resulting in more efficient learning.

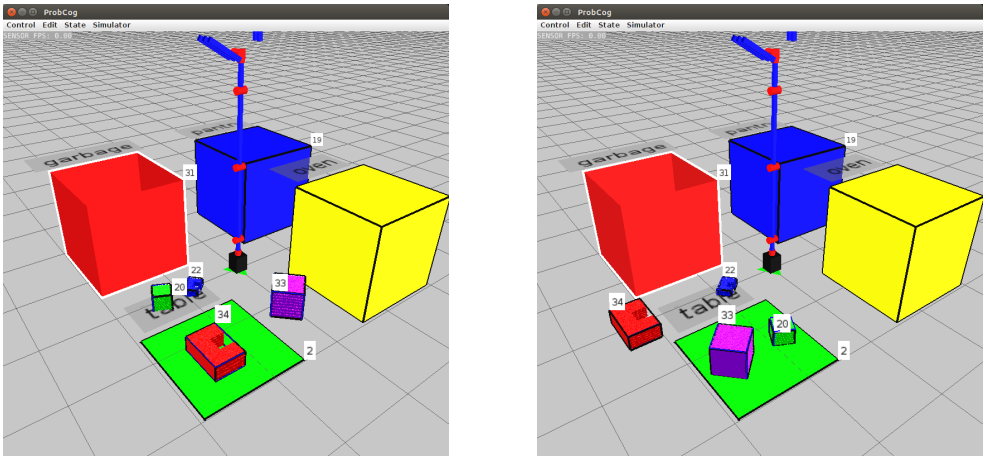

Figure 7.3: Initial and final environment states for learning *set the table task*.

7.4.2 Mixed-Control of Learning

The interaction model does not impose a strict order on how Rosie is taught new concepts. This gives flexibility in how the instruction is structured and who takes the onus of learning. The instructor can choose to control how and when Rosie learns. The instructor can teach basic concepts before teaching complex concepts that require knowledge of the basic concepts. However, the expert may not know or remember the state of Rosie's knowledge. In situations when the learner is learning complex concepts but lacks required knowledge of basic concepts, it will take the initiative and guide the interaction to acquire the relevant basic concepts first.

We taught Rosie the *set the table* task with two instructional strategies: instructor-controlled and agent-controlled. *Set the table* task requires that the *purple object* be placed *on* the *table* and the *green object* be placed on the *the right of the purple object.* The initial and final states of teaching the task are in Figure 7.3. The dialog trace in Figure 7.4 shows the instructor-controlled strategy in which the instructor teaches the basic concepts and then combines them to teach the *set the table* task. The dialog trace in Figure 7.5 shows the agent-controlled strategy where while learning *set the table* task, Rosie guides the interactions to basic concepts.

```
Mentor: this is red
Agent: What kind of attribute is red?
Mentor: a color
Agent: Okay.
Mentor: this is green
Agent: What kind of attribute is green?
Mentor: a color
Agent: Okay.
Mentor: this is blue
Agent: What kind of attribute is blue?
Mentor: a color
Agent: Okay.
Mentor: this is purple
Agent: What kind of attribute is purple?
Mentor: a color
Agent: Okay.                                          [learns 'purple']
Mentor: the red object is on the table
Agent: Okay.                                              [learns 'on']
Mentor: the blue object is right of the green object
Agent: Okay.                                       [learns 'right of']
Mentor: place the blue object on the table
Agent: This is a new task for me. What is the goal of this task?
Mentor: the goal is the blue object on the table
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: pick up the blue object
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: put the blue object on the table                  [learns task]
Agent: Test me or give me another task.
Mentor: set the table
Agent: This is a new task for me. What is the goal of this task?
Mentor: the goal is the purple object on the table and
        the green object right of the purple object
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: place the purple object on the table
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: place the green object right of the purple object [learns task]
Agent: Test me or give me another task.
```

Figure 7.4: Human-agent interaction trace for instructor-controlled teaching strategy (simplified).

```
Mentor: set the table
Agent: This is a new task for me. What is the goal of this task?
Mentor: the goal is the purple object on the table and
        the green object right of the purple object
Agent: I don't know the preposition right-of. Please teach me with examples
Mentor: the green object is right of the blue object
Agent: What kind of attribute is green?
Mentor: a color
Agent: I don't see the green object. Please teach me to recognize one
Mentor: this is green
Agent: What kind of attribute is blue?
Mentor: a color
Agent: I don't see the blue object. Please teach me to recognize one
Mentor: this is blue                                          [learns 'blue']
Agent: I don't see the green object. Please teach me to recognize one
Mentor: this is green                                        [learns 'green']
Agent: I don't know the preposition on. Please teach me with examples
Mentor: the red object is on the table
Agent: What kind of attribute is red?
Mentor: a color
Agent: I don't see the red object. Please teach me to recognize one
Mentor: this is red                                            [learns 'red']
Agent: What kind of attribute is purple?
Mentor: a color
Agent: I don't see the purple object. Please teach me to recognize one
Mentor: this is purple                                      [learns 'purple']
Agent: OK                                                       [learns 'on']
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: place the purple object on the table
Agent: This is a new task for me. What is the goal of this task?
Mentor: the goal is the purple object on the table
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: pick up the purple object
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: put the purple object on the table
Agent: I cannot figure it out.
Agent: How do I proceed?
Mentor: place the green object right of the purple object     [learns task]
```

Figure 7.5: Human-agent interaction trace for agent-controlled teaching strategy (simplified).

In the instructor-controlled strategy, the instructor is required to predict what concepts (such as the meaning of words *red*, *green*, *on*) will be useful in teaching the task and introduce these concepts early on. In the agent-controlled strategy, the instructor dives into teaching the task itself. Rosie focuses the dialog on different aspects of the task based on the state of its knowledge. Interaction in both teaching strategies encodes the same information and results in acquisition of same concepts.

7.5 Summary and Discussion

In this chapter, we described how Rosie maintains a mixed-initiative interaction with a humans instructor. The interaction model is based on Grosz and Sidner's 1986 collaborative discourse theory that posits that all discourse occurs in pursuit of intentional goals. The interaction model is impasse-driven - whenever the Rosie is unable to make progress on a task, it can ask relevant questions and change the state of interaction. This tailors the human-agent interaction to what is required by Rosie to successfully execute a task (D7). The discourse segments organize human-agent dialog in contiguous chunks of events that are aligned with task subgoals and information gathering acts. The state space of interactions is defined over joint space of utterances, actions, and learning. The interaction model incorporates non-linguistic actions and learning for reasoning about the interaction state. The model is integrated with comprehension and learning. Both processes influence how human-agent interaction progresses. The interaction provides useful information for effective reasoning in both processes (D8).

Both the instructor or Rosie can initiate a segment on the interaction, focusing the conversation on various aspects of the task (D9). The instructor does so by asking questions from Rosie or asking it to perform a task. The instructor can also provide examples (*this is red* while pointing to the object) without any prompts from Rosie. Rosie initiates new segments to gather information about the task in order to make progress on it. The

interaction model and the learning paradigm used does not impose strict order on how instructional dialog progresses. The model allows the instructor to have varying degrees of control over learning.

The interaction model proposed here makes several simplifying assumptions. Every utterance (except questions) from the instructor is interpreted and treated as a training example which eventually results in a change in Rosie's knowledge state. In more natural interactions, this assumption may not hold true. Participants may converse to understand how other agents interpret the situation, to model how their collaborators think, to alleviate perceptual difficulties, and for several other reasons. Reasoning about these aspects of collaborative interaction is not incorporated in our model but will inform our future efforts. Another assumption is that the instructor is always correct. An ideal learner should be critical of the input it receives and analyze the correctness of instructor's utterance using its own knowledge of the task. In situations where the instructor is naive, cannot model how Rosie learns, or cannot completely observe the environment, the quality of instruction may decrease. Rosie should detect these situations and take initiative in exposing its knowledge state or describing the scene. This requires research into a more expressive interaction state representation and conversational heuristics. The model does not incorporate non-linguistic actions from the instructor. Although, this may be true for most SII scenarios, this assumption may not hold in collaborative task execution where both participants can manipulate the shared environment and learn from each other.

Chapter 8
Learning Goal-Oriented Tasks

The task-oriented analysis in Chapter 5 suggests that several tasks undertaken in domestic or kitchen environments can be characterized as achieving a goal state through execution of a sequence of actions. For example, for a task such as *set the table* the agent must instantiate a goal of achieving a set of spatial predicates (including *fork is on the right of the plate*) and execute a series of object manipulation actions to achieve it. Several of these tasks (such as *serve dinner*) can be hierarchically decomposed into subtasks (*make dinner*, *set the table*, etc). To learn these tasks, the agent must not only learn the goal definition and how to recognize goal achievement from sensory data, but also learn an execution policy defined over subtasks and actions that achieves the goal in the environment.

Learning from demonstration (LfD) approaches (Argall et al., 2009; Chernova and Thomaz, 2014) have recently gained prominence in the robotics community as a way of allowing naive human users to teach new tasks and actions to a robot. Common LfD approaches rely on traces obtained through teleoperation or kinesthetic training. Using regression-based methods, these traces can be used to directly approximate policy. Although kinesthetic training is useful in learning *primitive* action-control policies (such as for object manipulation), it is unsuitable for learning complex tasks such as those characterized earlier. It does not capture abstractions such as subtasks and the corresponding transition models that are required for reasoning about and learning goal-oriented tasks (Grollman and Jenkins, 2010). It usually requires many examples in order to induce the intended hierarchical control structure (Allen et al., 2007). Moreover, the representations

are task-specific and are not amenable to transfer to structurally similar tasks (Chao et al., 2011).

In this chapter, we explore an alternative and complimentary approach for interactively learning new tasks (Mohan and J. Laird, 2014) based on explanation-based methods for learning and generalization (*EBG*: Mitchell et al., 1986; *EBL*: DeJong and Mooney, 1986; *Chunking*: J. Laird et al., 1986). In the following sections, we give an overview of explanation-based learning methods (Section 8.1), formulate the problem of learning goal-oriented tasks from explanation-based methods (Section 8.3), describe our approach (Section 8.4), present empirical results (Section 8.5) and analyze the degree to which our approach satisfies the SII interactive learning desiderata (Section 8.6).

8.1 Explanation-based Generalization

Explanation-based methods provide a principled way to exploit domain-knowledge for supervised learning. EBL methods offer several advantages over regression-based learning methods. They work over relational representations that not only allow the agent to reason about the structure of the goal and how to achieve it but are also useful in linguistic communication making them a suitable learning paradigm for SII. These methods are knowledge-intensive and exploit the agent's domain knowledge to deduce generally applicable knowledge from very sparse data examples (D14). The learning efficiency results from a key insight that it is possible to form a justified generalization of a single positive training example if the agent can explain why this is a positive example. Explanation-based methods elegantly combine *analytic* evidence provided by inference over domain knowledge with *empirical* evidence provided by the training data. Mitchell et al. (1986) formulate the explanation-based generalization (EBG) problem as:

Given:

- *Target concept.* A definition of the concept to be learned.

- *Training example.* An example instance that satisfies the definition of the target concept.
- *Domain theory.* A set of domain knowledge rules.
- *Operationality criterion.* A set of predicates specifying the form in which the learned concept definition must be expressed.

Determine:

- A generalization of the training example that is sufficient for the definition of the target concept and that satisfies the operationality criterion.

The EBG method accomplishes this goal in two steps.

- *Explain.* Construct an explanation using a general inference engine and the rules of domain theory to prove how the training example satisfies the goal concept.
- *Generalize.* Determine a set of sufficient conditions under which the explanation structure holds, expressed in terms of the operationally criterion.

8.2 Our Approach

Prior approaches in interactive learning such as LfD and interactive reinforcement learning address acquisition of execution policy. However, learning a new task from scratch requires acquiring other kinds of knowledge including goal definition and recognition, task pre-conditions, and task models. Previous methods assume that this knowledge is pre-encoded in the agent. In this chapter, we propose an interactive task learning method based on EBL that learns various aspects of task knowledge from interactions. The proposed method satisfies several desiderata for SII identified previously.

- **Multi-method**(D10). In order to learn various aspects of task representation, the proposed approach not only uses examples explicitly provided by the instructor but also deduces some task aspects. The method also relies on data-driven learning to associate values to implicit task parameters.

- **Assimilative**(D11). The proposed learning approach is *impasse*-driven. The failures in accessing relevant knowledge during task execution results in meta-cognitive analysis, interaction, and learning. Consequently, the learning is tailored to what Rosie needs for task execution. The impasses and the subsequent analysis provides context to how the next instruction should be processed and be incorporated in a comprehensive representation.

- **Multi-task**(D12). We show that our approach can learn different types of tasks as characterized in Chapter 5.

- **Fast**(D14) **Generalization**(D13). The reliance on relational representations and EBG for learning guarantees quick generalization of learning. We show that with few examples, Rosie can learn a task representation that applies to entire task variation in our domain.

- **Transferable**(D15). The hierarchical representation when combined with EBG lends itself to transfer of structural and policy information across similar tasks.

- **Active**(D16) and **Online**(D17). Rosie takes initiative in its own learning and actively asks for information when it lacks knowledge to progress further. The learning is online and occurs during performance. Consequently, Rosie can be taught several tasks without having to stop it in order to expand its knowledge as is common with other data-driven task learning approaches Tellex et al. (2011).

8.3 Formulating Task Learning as EBG

We now formally construct the problem of learning tasks with situated instruction. Our implementation is based on chunking in the Soar architecture (J. Laird et al., 1986) which is closely related to Mitchell et al. (1986).

8.3.1 Assumptions

As explained earlier in Section 4.2.2.3, Rosie's beliefs about its current state $s \in S$ are encoded as a set of relational predicates P defined over the set of objects O. We assume that Rosie is pre-encoded with a set of primitive behaviors or actions A that can be executed in the environment and with action models that predict the effects of their execution.

8.3.2 Formulation

We formulate the task learning problem by specifying the components of EBG as follows.

- *Target concepts.* The elements of task representation (identified in Section 5.3.4 and reproduced below) are the target concepts that have to be learned. Although our representation bears similarities with HTNs (Erol et al., 1992) and MAXQ (Dietterich, 2000) hierarchies, there are important differences. Control knowledge in HTNs is encoded as conditional actions whereas in our representation it is encoded as a state-sensitive policy. Encoding it as a policy allows Rosie to robustly operate in dynamic environments where the state may change independent of agent's actions without extra planning. In contrast to our representation, typical MAXQ hierarchies operate in propositional domains.

- *Training examples.* Examples to the learning algorithm are either obtained from the instructor or deduced by Rosie. Concepts such as *goal elaboration* are learned from examples by the instructor. Examples for concepts such as *availability condi-*

tions are deduced by Rosie. Examples for *policy* are either obtained from exploring the possible solution paths or by prompting the instructor for situated examples. By generating and deducing examples for EBL, Rosie takes some onus for its own learning (D16).

- *Domain theory.* The domain theory for task learning consists of the pre-encoded and acquired task models in addition to domain general rules that assert or verify truth-values of domain predicates.

- *Operationality criterion.* All concepts learned contain environmental state predicates or task operators. The environmental predicates are directly grounded in Rosie's sensory data. The task operators can be hierarchically decomposed to primitive action operators grounded in robot's control policies and functional manipulation of the environment.

8.4 Interactive Task Learning

Our design tightly couples execution of the task with exploratory and instructional acquisition through *impasse-driven* learning. When Rosie is give a task command, it uses all its knowledge (syntactic, structural, and procedural) to generate a grounded task operator and to instantiate a policy that can be executed in the environment. While trying to interpret and execute a novel task, it often reaches impasses or knowledge-retrieval failures and cannot progress further. When an impasse arises in Soar, it automatically creates a substate. The goal of the substate is to analyze and resolve an impasse. This allows Rosie to reason about why the impasse occurred and generate a relevant strategy for resolution. In some impasses, it formulates a question for the instructor and in others, it explores possible solutions using its domain models. Often in human-controlled interactive learning, such as learning by demonstration, the onus is on the human user to provide good examples so that the agent can acquire general hypotheses. In contrast, with

our approach, the instructor can rely on Rosie to initiate interactions when needed. This learning paradigm biases acquisition towards knowledge that is required by Rosie for task performance and it can speed up learning. The tight coupling of task performance with knowledge acquisition makes our learning paradigm online (D17). Substate reasoning in response to failures allows Rosie to monitor its knowledge state and guide its learning (D16) by asking relevant questions.

In the following sections, first, we describe how known tasks are executed and then how new tasks are learned. We use the task verb *store* as an example for our explanations and refer to Figure 8.1. For this task, we assume that the goal of $store(o)$ is to place any object o in the implicit location $pantry$ regardless of its perceptual attributes. Later in Section 8.4.3, we explain how associative default values for implicit parameters are learned and applied for task execution.

8.4.1 Executing a Known Task

On receiving a task command such as *store the red cylinder*, Rosie grounds it (explained in Chapter 6) to generate a task instantiation. Let this task instantiation be t. Once the task operator has been instantiated, it is processed in the following steps (as in Algorithm 3).

1. *Elaborate goal predicates.* If the goal elaboration operator for task t exists, it appends the instantiated goal predicates to the task operator (line 2, Algorithm 3). For the task $store(o)$, the goal predicates consist of $predicate(o, k2, o_2) \wedge closed(o_2)$ where $k2$ is the spatial composition that corresponding to preposition *in*, and $o2$ is the location *pantry*.

2. *Propose and select task operator.* If the availability conditions of the task operator match the current state description, the task operator t is proposed and is selected for execution. For $store(o)$, the availability conditions include predicates $block(o)$, $location(o_2)$, $not\text{-}predicate(o, k2, o_2)$, $not\text{-}closed(o_2)$.

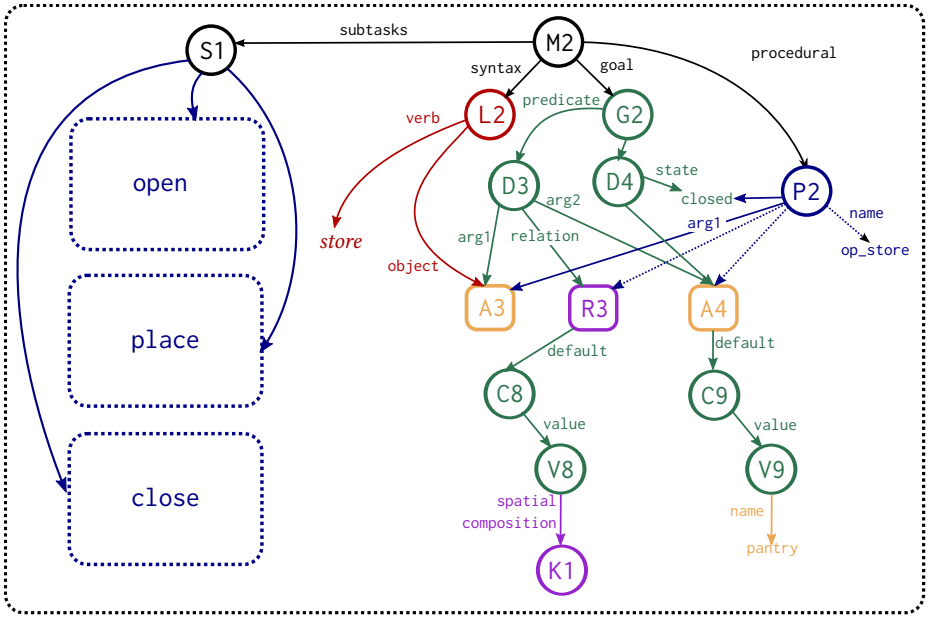

Figure 8.1: Semantic representation of *store*.

Algorithm 3 Executing a task.

procedure execute(state s, task t)
  if $d \leftarrow P^{t_G}$ **then**
    if $(P^{t_A} \subseteq s)$ **then**
      while $((d \subseteq s) \neq true)$ **do**
        $C \leftarrow c | available(s, c)$ & $c \in subtasks(t)$
        if $(\pi(s) \rightarrow c \in C)$ **then execute**(s, c)
        else
          if search$(s, d, C, \pi) = false$ **then subtask-query**(t)
  else
    $g \leftarrow$ **retrieve**(type $goal$, task t)
    if exists(g) **then**
      $d \leftarrow$ **instantiate**$(g, \{o_1, o_2, ...\})$
    else goal-query(t)

3. *Apply the task operator.* The subtask availability conditions determine if the primitive actions or subtasks can be applied in a state while executing t (Algorithm 3: line 5). Given the current state and the set of actions/subtasks that can be executed in that state, the policy suggests an action or subtask, which is then applied (Algorithm 3: line 6). The task execution is terminated when the desired state is reached (Algorithm 3: line 4). For $store(o)$, the subtasks are $open(o_2)$, $place(o1, k2, o_2)$, $close(o_2)$.

If the task command contains a verb that is novel to ROSIE, it attempts to learn the verb in the following stages. Assume that the verb *store* in *store the red cylinder* is a new verb.

8.4.1.1 Initialize TCN template

While grounding the command to generate a task instantiation, ROSIE tries to find a TCN associated with *store* in its semantic memory. This results in a failure which indicates that *store* is a new verb. This is an opportunity to learn a new task and ground the verb *store*. First, ROSIE extracts the general syntax used in the task command. From *store the red cylinder*, it extracts `verb:store object:<o>`. This is stored in the semantic memory (subgraph at `L2` in Figure 8.1). It, then, stores a new map node `M2`, connects it to `L2`, and creates the `subtasks` (`S2`), `goal`(`G2`), and `procedural`(`P2`) nodes. These nodes will be populated as the learning progresses. After the TCN template has been stored in memory, ROSIE appends to the `procedural` node a new task operator `op_store` that has one argument `arg1` with a constraint on its instantiation to values in the slot `A3`. After this new task operator is stored in semantic memory, the grounding function g_t successfully generates a task instantiation.

```
1.  Instructor: Store the red cylinder.
2.  Agent: What is the goal of the task?
3.  Instructor: The red cylinder is in the
    pantry and the pantry is closed.
4.  Agent: What action should I take next?
5.  Instructor: Open the pantry.
6.  Agent: What action should I take next?
7.  Instructor: Place the red cylinder in
    the pantry.
8.  Agent: What action should I take next?
9.  Instructor: Close the pantry.
10. retrospective explanation.
```

```
1.  Instructor: Store the red cylinder.
2.  Agent: What is the goal of the task?
3.  Instructor: The red cylinder is in the
    pantry and the pantry is closed.
4.  Agent: What action should I take next?
5.  Instructor: Open the pantry.
6.  Agent: What action should I take next?
7.  Instructor: Place the red cylinder in
    the pantry.
8.  retrospective explanation.
```

Figure 8.2: (left) Interactions for learning *store* with exploration depth $K = 0$. (right) Interactions for learning *store* with exploration depth $K = 2$.

8.4.2 Learning a New Task

8.4.2.1 Learn declarative goal definition

After instantiating the task operator, Rosie attempts to execute it (Figure 8.3: line 1). However, it is unable to generate the goal predicates (Algorithm 3: line 2, Figure 8.3: line 4) because it has not learned the goal elaboration rule yet. This results in a state-no-change impasse (Figure 8.3: state S3) in which Rosie queries its semantic memory for a description of the task goal (Algorithm 3: line 9). This is the first time Rosie is executing this task, therefore, the memory does not have a description of the goal. To learn the goal definition, Rosie asks a goal question (Figure 8.2(left): line 2, Algorithm 3: line 5, Figure 8.3: line 5).

The instructor replies with a description of the goal state for the ongoing task (Figure 8.2 (left): line 3). Rosie uses a grounding function g_G to generate an interpretation of the goal. The goal definition contains two predicates: a relation predicate between the objects referred to by NPs *the red cylinder* and *the pantry*; and a functional state predicate indicating that the object referred to by *the pantry* must be *closed*. Two goal predicate definitions (rooted at nodes `D3` and `D4` in Figure 8.1) are created in memory. The NP *the red cylinder* is common to both the task command and the goal definition. This indicates a constraint that `arg1` of the relation predicate should be instantiated with the explicit

Figure 8.3: Simplified Soar trace for interactive task execution.

parameter in the task command. This constraint is stored as the edge `arg1` between `D3` and the slot node `A3`. Other elements of the goal definition - the relation referred to by the preposition *in* and the object referred to by the NP *pantry* are implicit task parameters and are stored as default values of slots `R3` and `A4`.

An alternative approach is to not describe the goal explicitly but let the agent induce the goal from an instance of task execution. In this case, the agent cannot take initiative in exploring potential solutions while learning task execution.

8.4.2.2 Learn goal elaboration rule

After storing the definition of the goal in semantic memory, Rosie operationalizes it. In S3 (Figure 8.3), it queries its memory for a goal definition and instantiates it to task parameters. The instantiated goal is appended to the task operator (Figure 8.3: line 7) and chunking compiles a goal elaboration rule of the form:

$$\texttt{if } executing(t, s) \texttt{ then } t.d \leftarrow P^{t_G} \tag{8.1}$$

Example for *store*:

```
if state s:  location o2
         o2: name pantry
         task: t
   task  t:  name op_store
             arg1 o1
then
         t:  desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
```

8.4.2.3 Learn termination rule

Next, Rosie learns the termination rule (line 8, Figure 8.3). It imagines the state s_h in working memory that will result on executing the task in the current state by adding the grounded goal predicates to the current state description (Figure 8.3, line 8). Rosie then uses a set of domain-general rules to verify that every predicate in the goal is true in s_h and that s_h is a valid instance of the task's terminal state. Chunking complies this verification into a termination rule that is of the form:

$$\begin{aligned}&\texttt{if } executing(t, s)\\ &\quad \wedge (\forall p1, p1 \in t.d \rightarrow \exists p2, equivalent(p1, p2) \wedge p2 \in s) \texttt{ then } achieved\text{-}goal(s)\end{aligned} \tag{8.2}$$

The termination rule for the *store* task is:

```
if task  t: name op_store
            arg1 o1
            desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
   state s: predicate p3
            predicate p4
         p3: arg1 o1
             relation k2
             arg2 o2
         p4: arg1 o2
             state closed
then
        s: achieved-goal s
```

A general rule terminates the task in any state that is marked a terminal state (Figure 8.3: line 31, 34).

8.4.2.4 Learn policy from exploration

The task policy is learned during two different stages. A part of the policy is learned during execution of a task through immediate explanation. If the task policy does not suggest any subtask/action at state s (Algorithm 3: line 5, Figure 8.3: lines 18, 22, 26), Rosie performs a recursive iterative-deepening search for the desired state to depth K (Algorithm 4). For the exploratory search in state s, the agent iterates through all available subtasks/actions by applying their models and analyzing the resulting states (Algorithm 4: line 8). During this search, if an action a is found to be useful in making progress towards the desired state (Figure 8.3: lines 27-32), chunking compiles a policy rule. Chunking collects all the predicates that were tested to apply the models and the termination rule (Algorithm 4: line 8). The left-hand side of the policy rule contains these predicates and the right-hand side contains the action a. This rule is added to the task policy.

Algorithm 4 Exploring the action space

1: **procedure search**(state s, desired d, actions C, policy π)
2: **for** ($k = 0, k < K, k++$) **do**
3: $a \leftarrow$ **explore**(s, d, k)
4: **if** $a \neq false$ **then return** $true$
5: **return** $false$
6: **procedure explore**(state s, desired d, depth $n = k$)
7: **if** $n = 0$ **then return** $false$
8: **for each** $(a | a \in C \wedge available(s, a))$ **do**
9: $s' \leftarrow M^a(s)$
10: **while** $(\pi(s') \rightarrow c \in C)$ **do** $s' \leftarrow M^c(s')$
11: **if** $(d \subseteq s')$ **then**
12: $F \leftarrow$ **collect-tested-predicates**
13: **add**$(\pi(s | F \subseteq s) \rightarrow a)$ **return** $true$
14: **else**
15: **if** (**explore**$(s', d, n-1) \neq false$) **then goto** 10
16: **return** $false$

If the desired state is not found at depth K or earlier (Figure 8.3: lines 19, 23), Rosie abandons exploration and asks the instructor for an action that it should take. The in-

structor replies with a subtask/action command (Figure 8.2: line 5) which is grounded and executed in the environment. Such exploration and interactions continue until the agent achieves the goal state in the environment. Rosie's episodic memory automatically stores the state (environmental and interactive) of Rosie for each step. This history is available for later inspection and learning the remainder of the policy.

8.4.2.5 Learn policy from instruction

After the instructor indicates or Rosie deduces that the instructed task execution is over and the goal state is achieved, the agent attempts to learn the remainder of the policy. It is learned through simulating task execution and retrospectively explaining the instructions (in Algorithm 5, Figure 8.4). To simulate the task execution, Rosie queries its episodic memory for the environmental state s when it asked for the first instruction. It begins by instantiating the task goal definitions in state s to generate the desired state d. Next, it proposes all the subtask/actions that were used in the instructed execution and exploration of the task. Rosie then recursively analyzes why the next instructed action a (obtained by looking up episodic memory) is useful in approaching the desired goal state. In each recursion, Rosie applies the model of the instructed action M^a on the state s to generate the subsequent state s' (Algorithm 5: line 8). If Rosie has learned policy for s' through its exploration, it is applied (Algorithm 5: line 9, Figure 8.4, line 54). If the goal is achieved in any subsequent state s' desired, a policy rule and a subtask proposal rule is learned (Algorithm 5: line 11,12, Figure 8.4: lines 55, 59) through chunking. It collects all the predicates that were tested to apply the models and the termination rule (Algorithm 5: line 10). The left-hand side of the policy rule contains these predicates F and the right-hand side contains the action a. This rule is added to the task policy. If the goal is not achieved, it recurses further (Algorithm 5: lines 14, 15, Figure 8.4: lines 42, 48) by looking up episodic memory for the next instructed action. The policy rule is of the form:

$$\texttt{if}\ executing(t, s) \wedge (\forall p1, p1 \in F \rightarrow \exists p2, equivalent(p1, p2) \wedge p2 \in s)\ \texttt{then}\ a \quad (8.3)$$

Algorithm 5 Explaining instructions retrospectively

1: **procedure explain-instructions**(time n, task t)
2: $s \leftarrow$ **query**(n)
3: $d \leftarrow P^{t_G}$
4: $C \leftarrow c|available(s, c), c \in subtasks(t)$
5: $a \leftarrow$ **retrieve-first-action**
6: **procedure exploit**(s, a)
7: $s' \leftarrow M^a(s)$
8: **while** $(\pi(s') \rightarrow c \in C)$ **do** $s' \leftarrow M^c(s')$
9: **if** $(d \subseteq s')$ **then**
10: $F \leftarrow$ **collect-tested−predicates**
11: **add**$(\pi(s|F \subseteq s) \rightarrow a)$ **return**
12: **else**
13: **if** $a' \leftarrow$ **retrieve-next-action then**
14: **if** (**exploit**$(s', a') \neq false$) **then goto**: 8
15: **return** $false$

A policy rule of the task *store* is below.

```
if task  t: name op_store
            arg1 o1
            desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
   task t2: name op_close
            arg1 o2
   state s: predicate p3
            predicate p4
            task t2
            task t
         p3: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state open

then
        s: prefer t2
```

Figure 8.4: Simplified Soar trace for retrospective instruction-aided learning.

8.4.2.6 Learn proposal rule

The task proposal rule (that encodes the availability conditions) is learned at the end of policy learning. State S11 (in Figure 8.4) is a state in which the *store* task can be successfully executed and therefore is an example instance of a state where *store* should proposed. Chunking complies a proposal rule of the form:

$$\texttt{if}(\exists p1, p1 \in F^t \wedge (\exists p2, equivalent(p1, -p2) \wedge p2 \in s)) \texttt{ then } available(s, t) \quad (8.4)$$

where F^t is the set of predicates that were tested to successfully execute the task t in the initial state. For the task *store*, Rosie learns this proposal rule.

```
if state s: not-predicate p3
            predicate p4
            task-command t
            block (o1)
            location (o2)
          t: name op_store
            desired d
            arg1 o1
          d: predicate p1
            predicate p2
          p1: arg1 o1
              relation k2
              arg2 o2
          p2: arg1 o2
              state closed
          p3: arg1 o1
              relation k2
              arg2 o2
          p4: arg1 o2
              state closed
then
         s: task t
```

8.4.2.7 Notes

The model for a task t is a critical component of the domain theory for learning a parent task. It is used for explaining why the sequence of actions either discovered through exploration or given by the instructor lead to goal achievement. In our formulation, an

explicit representation of the model is not required. The effects of task t in a state s can be predicted by simulating the execution of t in s through applying its policy. This is sufficient for learning a novel parent task. Similarly, the application rules for various tasks are not learned but implemented by hierarchically executing the subtasks.

8.4.3 Learn Associative Default Values

In the previous sections, we described an interactive explanation-based formulation of learning hierarchical goal-oriented tasks. For the $store(o)$ task, we assumed that the implicit location $pantry$ applies to any object o. However, this assumption leads to incorrect behavior. Consider as an example the verb *put* in the chores dataset. It is used in the following two alternations:

1. *Put the utensils in the dishwasher.*
2. *Put away the books.*

In the second alternation, the thematic role of *location* is left unexpressed and is an implicit parameter of the *put* task. Presumably, the implicit location for *books* is the *shelf*. Using this implicit location generally with any object will result in incorrect execution of the task command *put away the groceries* for which the implicit location is unlikely to be the *shelf*. This suggests that the generalization strategy adopted in previous sections is aggressive. This results from only considering object state and affordances (available through proposal and application rules of actions and subtasks) in constructing explanations. Although this information is useful in learning what tasks are afforded in the current state and how they can be executed, it is insufficient to learn correct generalizations for tasks with implicit arguments.

In addition to physical affordances of a state and objects, a critical aspect of domestic environments is the semantic organization of a home or a kitchen. Even though *grocery* items *afford* to be placed on the *shelf*, they are not usually placed there in typical human homes. To align task representations with what human users intend and expect, such

semantics should be incorporated in the task structure. A key insight from the chores dataset is the selection and assignment of implicit parameters of verbs critically depends on the perceptual or semantic attributes (or classification) of the explicit parameters. For the verb *put* above, the semantic category of the explicit *theme* determines what *location* (*book*:*shelf*) it should be moved to. For the verb *load* in *load the dishwasher* and *load the washing machine*, the *location* determines what *theme* (*dishwasher*:*utensil*) applies.

Even though it is critical to task representation and learning, the issue of implicit task parameters has not been addressed in the prior work. The following sections propose a simple inductive concept learning approach to acquiring implicit task parameters.

8.4.3.1 Inductive Concept Learning for Goal Elaboration

We formulate the problem of associating correct implicit arguments with the task operator based on how explicit values are characterized as a concept learning problem. The implicit parameters are elaborated by the goal elaboration rule (Section 8.4.2.2) which is learned through chunking by deliberately instantiating the declarative definition of the goal (in semantic memory; Figure 8.1, subgraph rooted at `G2`). The declarative goal definition contains constraints on how predicates (`D3`, `D4`) and their arguments (`A3`, `A4`, `R3`) are instantiated given a grounded task command. To learn associations between explicit and implicit arguments, additional constraints are learned. Consider Figure 8.5. Node `C2` suggests a default value for slot node `A4` in case when values for `A4` are not explicit in the grounded task command. It encodes that if `arg1` (explicit argument) of task operator `op_store` contains perceptual attributes `color:red`, `volume:closed`, `shape:arch`, `size:small` then any object on the scene that has the perceptual attributes `name:pantry` and `volume:container` should be used as a value for slot `A4`. Similarly, the default value of the relation node `R1` is constrained by the perceptual classifications of objects in `arg1` and `arg2` in the instantiation of predicate `D3`. In our formulation, these constraints are incrementally learned from instructions using the find-S algorithm for learning version

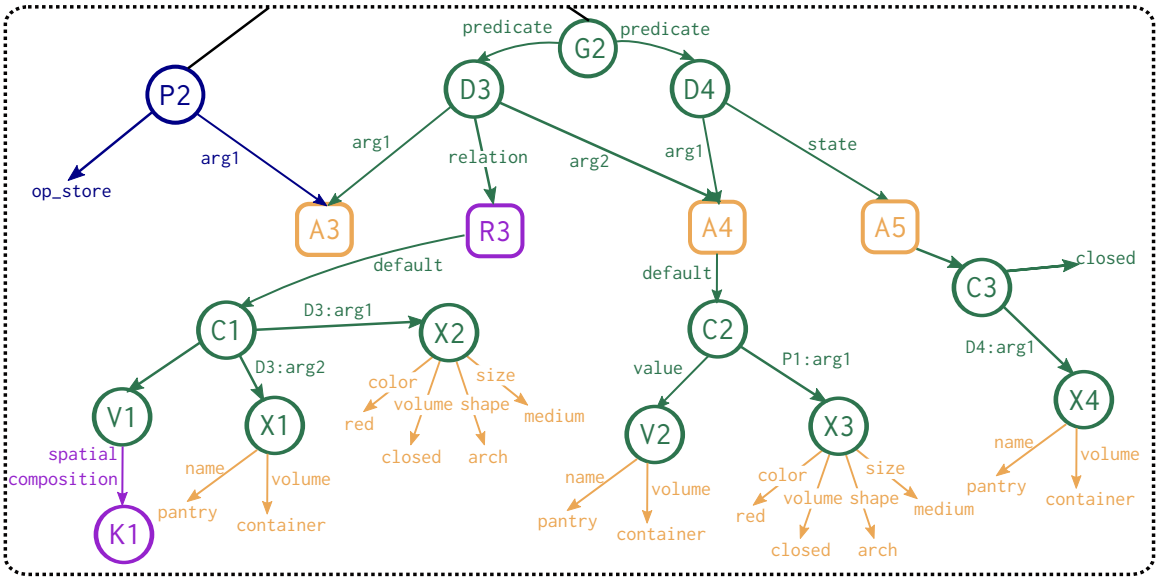

Figure 8.5: Associative default values for *store*.

spaces as described below.

Assume that a novel $store(o)$ task has two variations. If the object o is of type $arch$, then it has to be put in ($r1$) the location $pantry$ ($l1$) and if it is of type $cube$, it has to be put on ($r2$) the $table$ ($l2$). If the implicit location is $pantry$, it has to be closed as well. Assume that the training environment has two arches - red (object $o1$) and blue (object $o2$), and two cubes - purple ($o3$) and green ($o4$).

8.4.3.2 Learning the most specific hypothesis

When Rosie is asked to execute $store(o1)$, it fails as it does not know what the task goal is. It asks a goal question. The instructor replies *the goal is the red object is in the pantry and the pantry is closed.* After grounding the goal description, Rosie extracts the most specific hypothesis about what the goal description could be. The goal consists of two predicates: a spatial relation predicate $p1 = (o1, r1, l1)$ and a state predicate $p2 = (l1, closed)$. $o1$ is the explicit parameter, whereas $r1$, $l1$, and $closed$ are implicit.

The hypothesis about predicate arguments are stored in semantic memory as follows:

- *relation predicate*. Relationships are expressed as a ternary predicate $(obj1, relation, obj2)$. If $obj1$ or $obj2$ are identified as implicit arguments, associative default values are added to their slot nodes. Associative default values for $obj1$, $obj2$ consist of a value node that captures their perceptual attributes (V2 in Figure 8.5) and a constraint node that captures all the perceptual attributes of the explicit argument of the task operator (X3 in Figure 8.5). The associative default values for $relation$ contain one value node that contains the relevant spatial composition (V2 in Figure 8.5) and two constraint nodes that contain all perceptual attributes of $obj1$ (X1 in Figure 8.5) and $obj2$ (X2 in Figure 8.5).

- *state predicate*: State predicates are expressed as binary predicates $(obj1, state)$. As in a relation predicate, if $obj1$ is an implicit argument, then the corresponding slot has a value node (V2 in Figure 8.5) and a constraint node (X3 in Figure 8.5). The associative default for $state$ contains a value node (`closed` in Figure 8.5) and a constraint node (X4 in Figure 8.5)

Once these constraints are stored, Rosie learns a goal elaboration rule:

```
if state s:  location o2
         o2: name pantry
             volume container
         task: t
   task  t:  name op_store
             arg1 o1
         o1: color red
             shape arch
             volume closed
             size medium
then
         t:  desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
```

which contains the appropriate constraints over objects. This rule represents the most specific hypothesis about the goal. After this, Rosie attempts to learn the task policy as described previously.

8.4.3.3 Detecting mismatch

When Rosie is asked to *store the blue arch* or *store the green cube*, the goal elaboration rule above does not fire because the perceptual attributes of the object corresponding to *the blue object* ($o2$) or *the green cube* ($o4$) do not match the constraints in the rule. Rosie cannot progress without a goal, therefore, it tries to deliberately instantiate a goal using the declarative goal definition and associative default values in its semantic memory. This attempt fails as the explicit argument ($o2$ or $o4$) here does not satisfy the constraints for default values of slot `R3`, `A4`, and `A5`. This failure indicates that associative default values are over-constrained and should be revised. To gather more information about how the goal should be revised, Rosie asks a goal question again.

8.4.3.4 Revising the hypothesis

On getting a new goal description, Rosie merges it with the declarative definition of the goal in its semantic memory as follows:

- *Relax constraints.* In our example, an *arch* is stored in the $pantry$. For the task command, *store the blue arch*, the goal description is *the goal is the blue arch in the pantry and the pantry closed.* This description aligns with the goal definition in semantic memory except that the associative default value is over-constrained. In order to merge the provided goal description, the constraints are relaxed by removing the perceptual attributes that are not common with $o2$ from associative default values for `A4` and `A3`. Updated constraints are shown in Figure 8.6.

  After relaxing constraints in semantic memory, Rosie learns a new goal elaboration

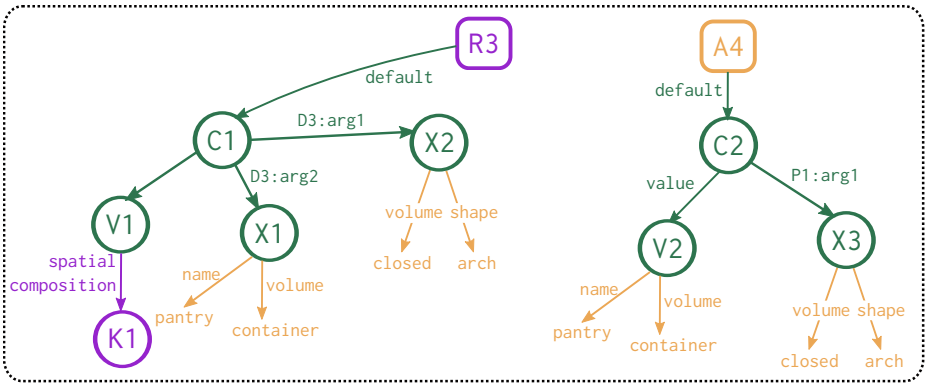

Figure 8.6: Relaxed associative default values for R3 and A4 in structural knowledge of *store*.

rule.

```
if state s:  location o2
         o2: name pantry
             volume container
         task: t
   task  t:  name op_store
             arg1 o1
         o1: shape arch
             volume closed
then
         t:  desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
```

This rule represents the most specific consistent hypothesis about the task goal given two examples.

- *Add new default value.* For the task command, *store the green cube*, the goal description provided is *the goal is the green cube on the table.* This description identifies a new implicit location and a new spatial relationship between the object and the location. Therefore, new default associative values (C3 and C4) are added to R3 and A4 (shown in Figure 8.7).

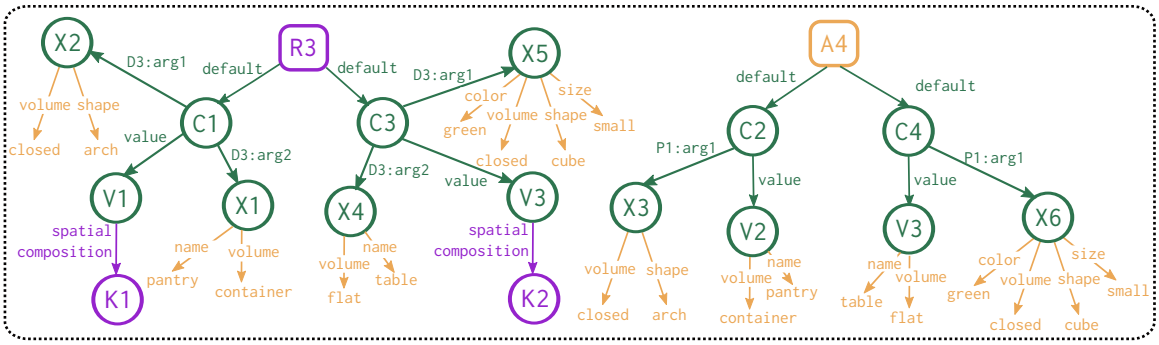

Figure 8.7: New associative default values for R3 and A4 in structural knowledge of *store*.

A corresponding goal elaboration rule (below) is learned.

```
if state s:  location o2
         o2: name table
             volume flat
         task: t
   task  t:  name op_store
             arg1 o1
         o1: shape cube
             volume closed
             size small
             color green
then
         t:  desired d
         d:  predicate p1
             predicate p2
         p1: arg1 o1
             relation k2
             arg2 o2
         p2: arg1 o2
             state closed
```

On getting further examples new default values are added or current constrains are relaxed based on how goals are described.

8.4.3.5 Notes

In our representation, the perceptual constraints only appear in the goal elaboration rule. Other rules - availability, policy, and termination - are dependent on how the goal is

instantiated. Consequently, as soon as a new goal elaboration is learned, other rules generalize implicitly without needing any more examples or instructions.

8.5 Evaluation

8.5.1 Comprehensiveness

Rosie can learn goal *achievement* tasks, where it acts to achieve a composition of goal predicates in the environment. A summary of the seven tasks taught to Rosie is in Table 8.1. The goals are composed of *state* and *spatial* predicates that eventually ground out to the functional and continuous state of the environment. The policy space for all tasks eventually grounds out to a set of primitive actions (including closed loop motor-control for object manipulation) that can be executed in the environment and an internal *wait-until* action that polls the environment for a state change (such as $cooked(o)$). It can learn both organizational (*place*, *stack*) and functional (*cook*) tasks.

Explicit parameters	*Primitive policy space*	*Goal description*
place (`obj`,`rel`,`loc`)	pick-up, put-down	`rel(obj,loc)`
move (`obj`,`loc`)	pick-up, put-down	`on/in(obj,loc)`
discard (`obj`)	pick-up, put-down	`in(obj,garbage)`
store (`obj`)	open, pick-up, put-down, close	`(in(obj,pantry) closed(pantry))` or `on(obj,table)`
cook (`obj`)	activate, pick-up, put-down, stop, wait-until	`in(obj,stove) cooked(obj)`
stack (`obj1`, `obj2`, `obj3`)	pick-up, put-down	`on(obj1,obj2) on(obj2,obj3)`
serve (`obj`)	activate, pick-up, put-down, stop, wait-until	`in(obj,stove) cooked(obj)`
set (`table`)	pick-up, put-down	`on(obj1,table)`, `right-of(obj2,obj1)`

Table 8.1: Learned tasks, parameters, policy space, and goals

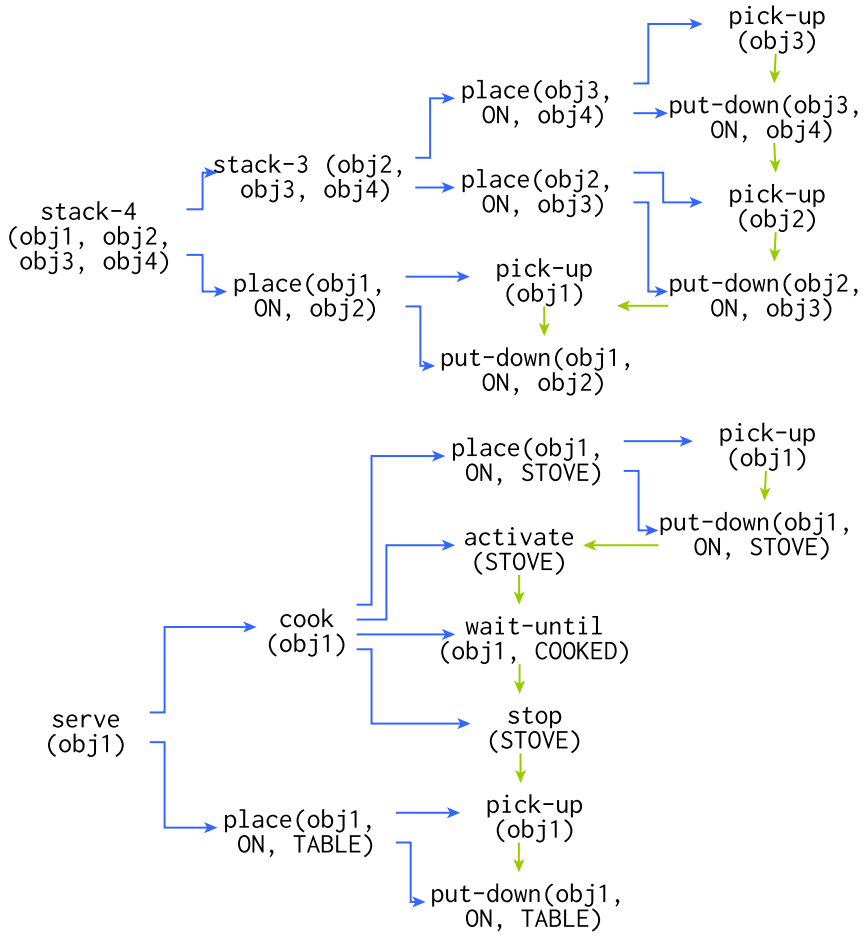

Figure 8.8: Learned hierarchical policies.

8.5.1.1 Hierarchical learning

The policy learned for task execution is flat if the instructions consist only of primitive actions or hierarchical if the instructions decompose the task into subtasks. For example, $serve(o)$ can be decomposed into $cook(o)$ and $place(o, on, table)$. $cook(o)$ can be further decomposed into its constituent tasks and actions. If Rosie does not know the subtasks, it attempts to learn it before learning the parent task. Examples of learned hierarchical policy are in Figure 8.8.

8.5.1.2 Implicit parameters

Rosie can learn tasks with *explicit* and *implicit* parameters. The learned *place* task is represented such that it can be used to achieve any known arbitrary spatial relationship between two objects and takes three arguments. These arguments explicitly identify all the information required to perform this task. A similar task *move* is defined for a specific spatial relationship on between two arguments if the second argument is not a container. Example, *move the red object to the table.* If the second argument is a container, the specific spatial relationship is *in.* Example, *move the red object to the pantry.* The relationships on, in are implicit parameters, are inherent to the *move* task, and depend on the explicit arguments. The learning paradigm can learn the following types of implicit parameters:

- *destination.* Tasks *store* and *discard* have implicit *destinations.*
- *instrument.* Task *cook* has an implicit *instrument* - stove.
- *theme.* Task *set the table* has implicit *themes* - the objects that have to be arranged on the table.

8.5.2 Generality

There are the following three types of generalization in our approach.

- *Relational abstraction.* Our learning paradigm works with relational state representations which abstract away the positional information in example instances.
- *Predicate selection.* The causal inference identifies the minimal and sufficient set of predicates from the complete state description that are required to apply the domain theory during learning. Consequently, the rules representing availability, termination, and policy apply in multiple states even though they have been learned from specific examples.

- *Object variablization.* The final type is *variablization* of objects and relations in task representations. We use the structure of interactions to inform which objects can be variablized away and what information should be retained for other objects. The objects and relations that are used in the task command (*explicit* parameters) are variablized away from example instances. For objects, relations, and states that do not occur in the task command (*implicit* parameters) but are used in goal description and task performance, the following strategies were considered.

  - *Variablize all objects.* An aggressive strategy suggests that all objects can be variablized. This strategy leads to over-generalization and tasks that have implicit parameters cannot be learned correctly and therefore, it was not considered.

  - *Learn constants.* The implicit parameters were incorporated in the task representation as constants.

  - *Learn associative default values.* The implicit parameters were selected during task performance based on the constraints and values in associative default values (as described in Section 8.4.3).

We conducted two experiments to evaluate the generality of our proposed method. The first experimental scenario consisted of four objects, four known spatial relations, and four locations. We conducted separate trials for learning flat execution for two tasks *place* and *cook*. A task trial consists of a series of episodes in each of which Rosie is asked to execute the task with randomly generated parameters. Each episode begins in an initial state obtained by assigning random states to locations $open/close(pantry)$, the arm ($hold/-holds(o)$), and arbitrarily placed objects on the workspace. The environment can be in 16 initial states and the objects can be in infinitely many locations. If the agent asked a child query during a training episode, it is given the relevant primitive action. An episode terminates when Rosie successfully executes the task in the environment. The

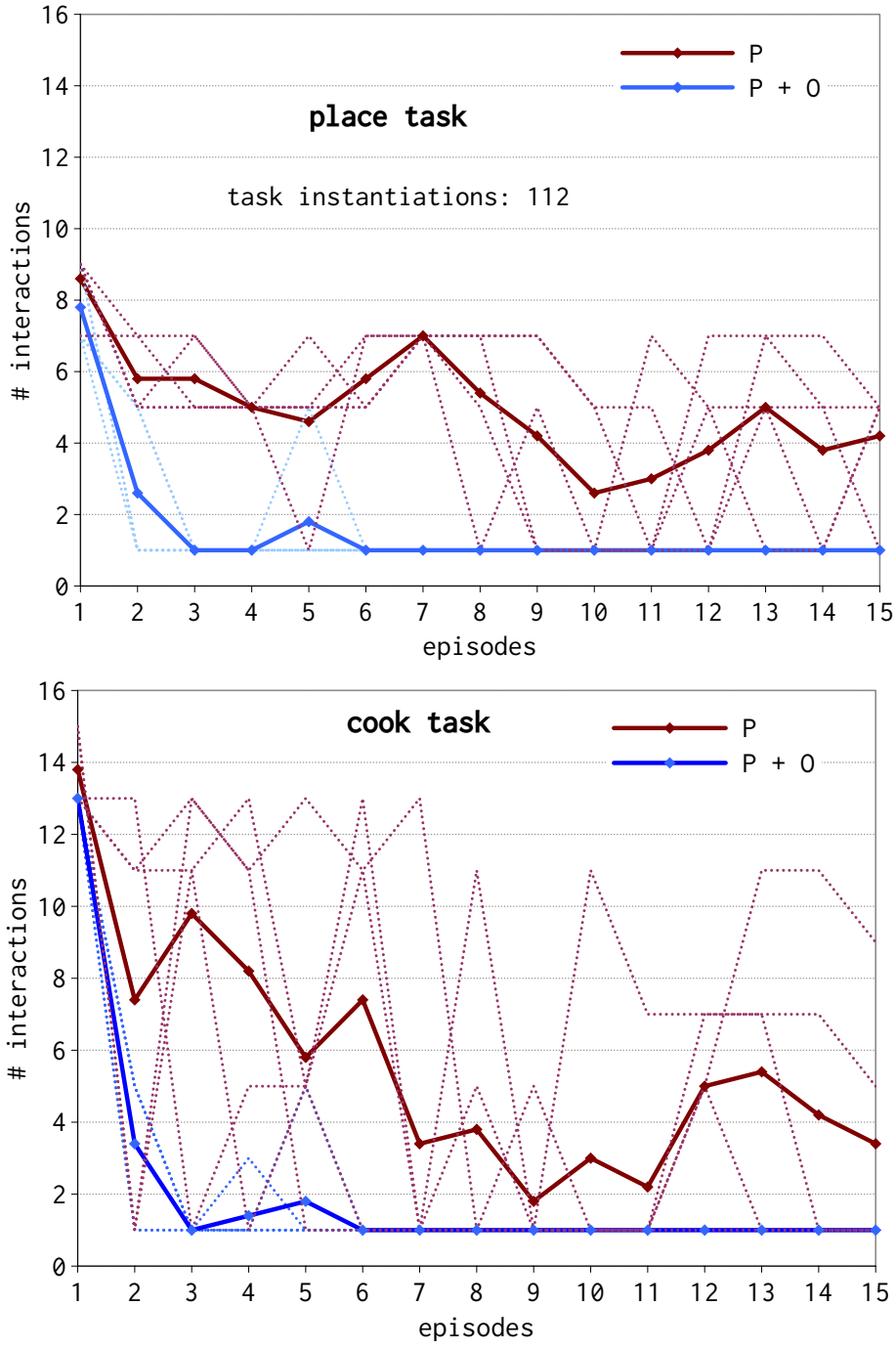

Figure 8.9: Procedural generalization during task learning.

exploration depth was set to 0 for this experiment.

A sample of the results generated from the experiment is shown in Figure 8.9. The graph shows the median number of interactions that occurred in every episode for executing *place* and *cook* over five trials (shown in dotted lines) for two variations of the learning algorithm. The first variation (red) only generalized through predicate selection. The second variation (blue) also variablized explicit parameters of the task. As expected, the majority of interactions occur during the first few episodes during which Rosie is trying to learn the task from interactions. The interactions drop as the trial progresses and Rosie learns to execute it without any instructions. The number of interactions for the second variation (blue) drop sharply after only a few episodes. This establishes that even though it has been trained on only a very small sample of the possible initial states (16) and task-command instantiations (112 for *place*, 4 for *cook*), it is able to learn representations that generalize to the complete space of command instantiations and initial states. The first variation (red) cannot generalize to the complete space of command instantiations as quickly but does generalize to the complete space of initial states. Both variations are insensitive to the specific positional information of the objects in the training instances as both use relational representations and both learn the correct policy. The data demonstrate that both predicate selection and variablization contribute towards general learning.

The second experiment evaluated if the two methods implemented for implicit parameters learned correct generalizations. The test environment contained eight objects that had four colors (*red*, *blue*, *green*, *purple*), two sizes (*medium*, *small*), and two shapes (*arch*, *cubes*). The environment had three locations. Pantry and garbage are containers and table is a flat surface. Following tasks were evaluated.

- *store [object]*. If the explicit object is an *arch*, it should be put in the pantry and the pantry should be closed. If the explicit object is a *cube*, it should be placed on the table. To teach *store*, task instantiations were created with every object and Rosie

was asked to perform the instances in random order. If it asked any questions, appropriate answers were provided.

- *move [object, location].* If the explicit location is a container, the explicit object is placed *in* the location. If the location is a flat surface, the object is placed *on* the location. To teach *move*, task instantiations were created with an object and a location selected randomly and Rosie was asked to execute them. If it asked any questions, appropriate answers were provided.

- *set the table.* A medium size object is placed on the table and a small object is place on the right of the medium object. To teach *set the table*, four pairs containing a small object and a medium object were created. Rosie was presented these pairs one by one and was asked to perform the task. If it asked any questions, appropriate answers were provided.

The results for learning implicit parameters with both methods are in Figure 8.10 which shows the number of interactions taken by each to learn these three tasks. Learning constants for implicit parameters implements an aggressive generalization strategy that assumes that the first example of the implicit parameter is the correct parameter for all instances. This results in incorrect task performance for *store* and *move* and failure to perform the *set the table* task in several instances because they have varying implicit parameters. Learning associative default values generalizes more carefully, beginning with a very specific hypothesis which is relaxed as it is given more examples. This requires more human-agent interactions in comparison to learning constants, but it is able to learn the task variations described above.

8.5.3 Transfer

Tasks in a domain may have similar structure, common subgoals, and overlapping policy. An ideal learner should be able to exploit the inter-task similarities while learning a new

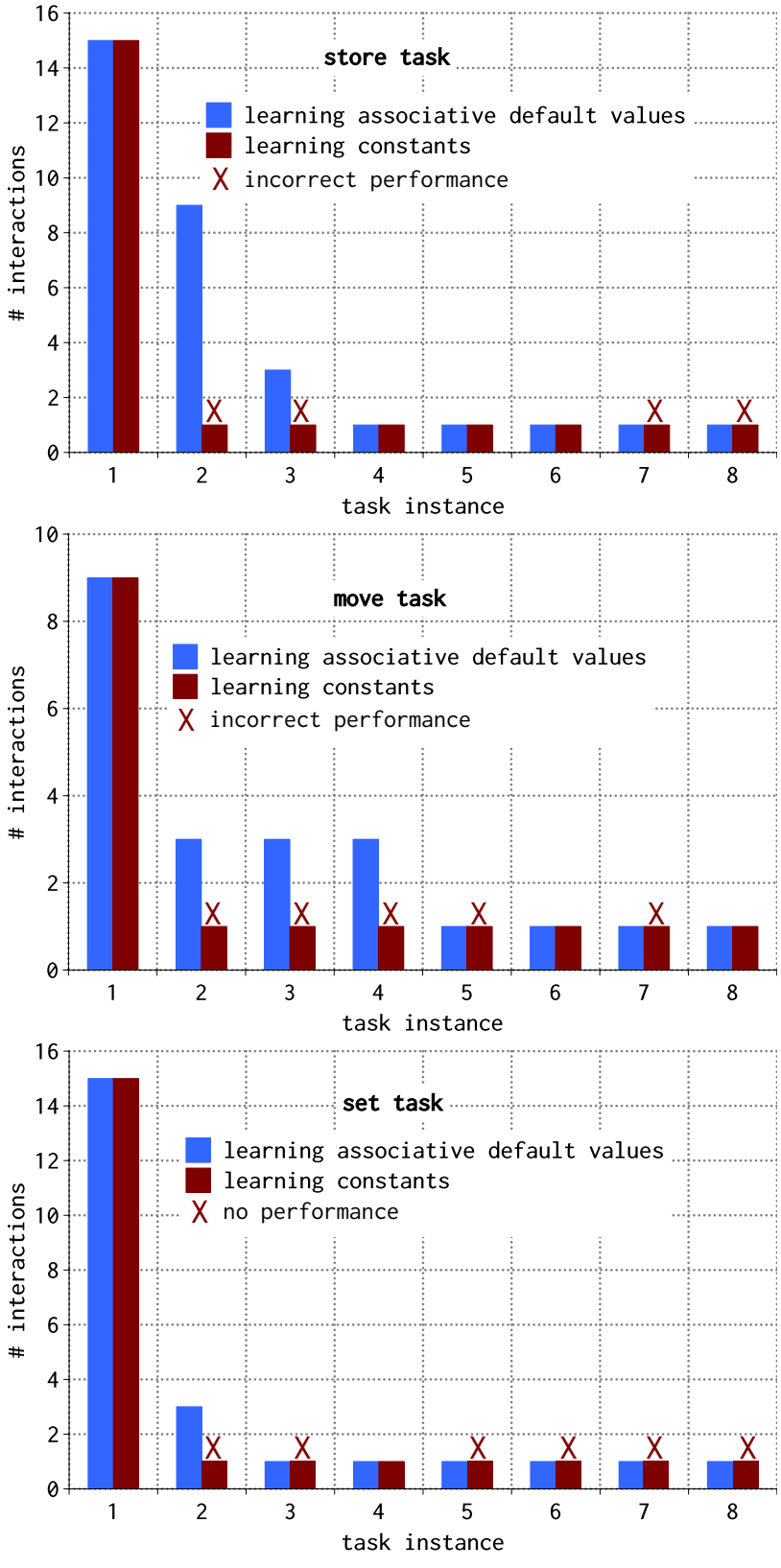

Figure 8.10: Conceptual generalization for learning implicit parameters of tasks.

task. For an interactive learner, the structure of interactions can play an important role in transfer of knowledge between tasks. Consider the tasks *store* and *cook* in our domain. Both of these tasks involve establishing a specific spatial relationship (in) between their parameters which is established by a policy over *pick-up* and *put-down*. This policy can be compiled in a *place* subtask through instructions and can be used to teach both *store* and *cook*, resulting in savings in interactions. This compilation is also useful in intra-task transfer in tasks such as *stack* that involves multiple *pick-up* and *put-down* compositions.

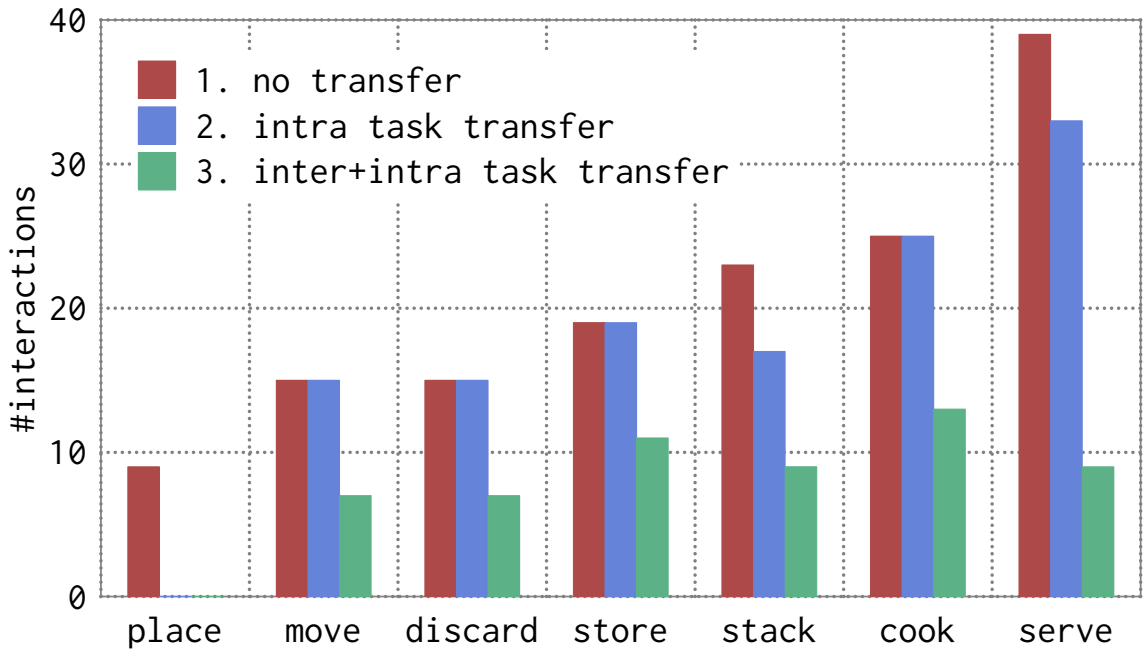

Figure 8.11: Instruction assisted transfer during learning.

Figure 8.11 shows how prior learning influences learning a new task. Rosie was taught seven tasks sequentially with three variations of the learning algorithm. The tasks were taught hierarchically by decomposing them into subtasks through instructions. In hierarchical learning, if the subtasks are known to Rosie, they are executed. If not, Rosie learns the subtasks before learning the parent task. In the first variation (red), Rosie's knowledge was reset after teaching each task (no inter-task transfer) and the learning of subtasks was turned off (no intra-task transfer). For the second variation (blue), learning

of subtasks was turned on (intra-task transfer). Finally, for the third variation (green) the knowledge acquired for each task was maintained allowing for inter- and intra- task transfer.

The *place* task is a policy defined over *pick-up* and *put-down* and has no subtasks. The *move*, *discard*, *store* tasks require a single *pick-up* & *put-down* composition along with other actions. If the *place* task is known before Rosie begins to learn these tasks, there are some savings in the number of interactions. There is no possibility of intra-task transfer. The task *stack* requires multiple executions of *pick-up* & *put-down* composition. Therefore, intra-task transfer is useful and saves some interactions. If *place* task is known prior to learning *stack*, further savings are achieved as the knowledge transfers from *place* to *stack*. Similar savings are observed in learning the *cook* and *serve* tasks when their subtasks are already known. The ability to transfer knowledge across tasks results in efficient learning as it significantly reduces the number of interactions required to learn a task.

8.5.4 Mixed-Initiative

Often in human controlled interactive learning, such as learning by demonstration, the onus of learning is completely on the human user. The human has to provide good demonstration traces that will result in learning at appropriate levels on generality. However, an ideal interactive learner must be active and play a useful role in knowledge acquisition. It should not completely rely on the human instructor, but instead use its knowledge of the domain to explore available options. Such active learning biases acquisition towards knowledge that is required by Rosie for task performance and reduces the load on the instructor. In our approach, Rosie uses its action/task models to explore the space of available actions to depth K to find a path to the desired state. Chunking complies deliberate exploration into a policy. If Rosie is unable to discover a path to the desired state, it abandons exploration and asks for guidance. During retrospection, the learning from agent-driven exploration and instruction-driven execution is integrated into a compre-

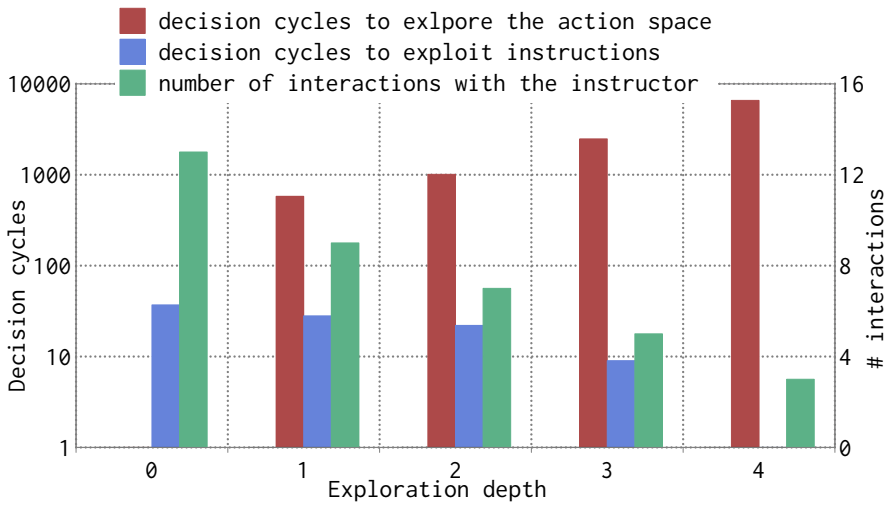

Figure 8.12: Learning *store* at different depths of exploration.

hensive policy.

Figure 8.12 shows the performance of the learning algorithm at different exploration depths for learning the store task in terms of the time spent in explorations (red bars as measured in Soar's decision cycles), the time spent in retroactive explanation (blue bars - decision cycles) and the number of interactions with the instructor (green bars). At depth 0, Rosie does not perform any exploration and relies completely on instructions. The entire time used for learning (blue bars) is spent on retrospectively explaining the instructions. As the exploration depth parameter increases, Rosie is more self-reliant, requiring fewer interactions, but spending more time exploring. At depth 4, it discovers the solution and does not ask any child-queries. Thus, the agent can solve simple problems on its own, only asking for help for difficult problems.

The results presented here illustrate that the proposed method can integrate instructor-driven interactive learning with self-driven exploration into a comprehensive task representation. A question useful from an HRI perspective is the specific value of the depth parameter d in the agent. Our experiments do not suggest any value the depth parameter should be assigned. This value should be determined through a HRI study. The claim we make here that our method can accommodate any d determined suitable for interac-

tive learning through further analysis and studies. A future research direction would be to extend Rosie so that it can explicitly reason about the expected cost and benefits of exploration and asking for help.

8.6 Summary and Discussion

In this chapter, we proposed an interactive variation of explanation-based generalization that is useful in learning the hierarchical, composable task representation proposed in Chapter 5 from specific examples of interactive task execution (D13). The task representations and the interactive learning paradigm allows Rosie to learn various types of tasks (D12). The learning paradigm is impasse-driven and allows Rosie to tailor the instructions according to its knowledge state (D16). It can take initiative in exploring different solutions to the goal. The interaction state (described in detail in Chapter 7) provides context for assimilating incremental instructions in a comprehensive schema (D11). The tasks we explored can be learned quickly (D14) and the knowledge learned for one task can be transferred to a structurally similar task (D15). The impasse-driven method proposed here works in real-time and allows Rosie to learn during task performance (D17).

There are several avenues for future research and exploration. One issue is our assumption that the instructor does not make any instruction errors. This may not hold for instructions for complex tasks in partially observable domains or novice instructors. A critical future direction of research is dealing with incorrect instructions. There are two important aspects to this issue. The first is detecting when the instruction is incorrect. The current implementation may provide some support for this. A failure to explain a sequence of instructions (detected through an impasse) suggests that either the goal was incorrectly described or the instructor missed actions critical for task performance. Additional reasoning, exploration, or verification questions to the instructor in the resulting substate can provide more information about the type of error. The second aspect is han-

dling incorrect data. In its current formulation, the learning paradigm induces a general task representation quickly from a single example. This may be dangerous as the one example may be an anomaly or incorrectly labeled or described. A potential solution is to gradually build confidence in task representations. The first task execution is guided through instruction. In a few later executions, the agent verifies if what it has learned is the correct way of reasoning about the task. Only, after it has built enough confidence in its task representations, does it perform the task autonomously.

Another issue is the expectation that the instructor is always available to respond to queries. An ideal learner would model the instructor and reason about the availability of the instructor and invest more resources in exploration, if the instructor is unavailable or unwilling. Finally, we are interesting in exploring the integration of instructional and experiential learning. In a dynamic world, the provided instruction may get outdated. The instructions should be interpreted as a template for acceptable behavior which guide agent's exploration of its environment and using its experience to modify and update task representations.

Chapter 9
Conclusions

In the previous chapters, we developed an approach to interactive task learning that relies on mixed-initiative human-agent instructional dialog - situated interactive instruction. In order to design a social learner, several computational challenges have to be met. This thesis presents an account of three of those challenges - situated comprehension, mixed-initiative interaction, and interactive task learning. The proposed methods have been integrated for end-to-end intelligent behavior in Rosie - an interactive task learned developed in the Soar. Rosie not only maintains an ongoing dialog about its environment but also reasons about communication and acquires domain knowledge from it.

To develop our approach, we proposed answers to the following questions:

- *How can verbs be grounded in task goals and execution knowledge?* Answering this question is critical to the design of interactive agents that not only communicate about tasks in natural language with their human collaborators but also can learn from such interactions. This question has been studied from several perspectives. While the research in vision has looked at labeling actions in videos (Siskind, 2001), HRI researcher have focused on generating control programs in response to commands (Bailey, 1997; Kollar et al., 2014). However, none has studied the relationship between verb semantics and task knowledge. This thesis takes initial steps towards aligning the meaning of verbs with various aspects to task representation. A major contribution of this thesis is a mixed-modality representation for grounding verbs in tasks.

In Chapter 5, we presented a preliminary semantic and task-oriented analysis of how people use verbs to describe domestic tasks. Motivated by the analysis, we proposed a mixed-modality representation of task verbs that encompasses their lexical, structural, and procedural aspects. The proposed representation encodes several components of verb semantics as described in VerbNet including expressed and unexpressed objects, semantic and state-driven selectional restrictions, and environmental predicates. It allows for hierarchical organization of task execution policies. We show that the proposed representation provides useful information in the linguistic task of situated comprehension of task commands. We also show how it can be realized in the memories of a general cognitive architecture. The representation is learnable and can be acquired from situated interactive instruction.

- *How can task commands be understood?* The challenge of situated comprehension of language has recently gained prominence in natural language processing (Liang et al., 2009; D. Chen and Mooney, 2011) as well as human-robot interaction (Cantrell, Scheutz, et al., 2010; Tellex et al., 2011; Chai et al., 2014). This thesis takes a significant departure from previous approaches that formulate situated comprehension as a parsing problem. The Indexical model formulates the problem of situated comprehension as a search over short and long-term knowledge for referents and composing them under syntactic and environmental constraints. The model integrates with knowledge acquisition and consequently, expands as Rosie accumulates knowledge about its world.

  Previous approaches have largely been silent on the role of non-linguistic contexts and background knowledge on comprehension, however, they play an important role in human language comprehension and generation. Non-linguistic contexts have a natural role in the Indexical formulation. They provide useful constraints over the search and composition which can reduce semantic ambiguities. We demonstrated that our model can effectively use perceptual, spatial, and task

knowledge with attentional contexts to reducing ambiguity in referring expression resolution. We also demonstrated that the model can exploit knowledge of task goals to handle commands that incompletely specify task arguments. Although the model proposed in this thesis is incomplete, it takes important initial steps for more human-like comprehension capabilities for intelligent agents.

- *How can task-oriented linguistic interaction be sustained?* Maintaining a task-oriented interaction is important for a taskable interactive agent and has been a research focus in the dialog community. Prior work on collaborative discourse theory (Grosz and Sidner, 1986; Rich and Sidner, 1998) has proposed a computational formalism for sustaining a task-oriented interaction. This thesis makes a minor contribution to the prior work by extending the prior formalism to support interactive learning. Our model is integrated with a comprehension and a learning module allowing Rosie to change the ongoing interaction if it fails to comprehend a sentence or to execute a task. The information so acquired informs knowledge acquisition. It captures the dialog context that is useful for constraining hypotheses about the meaning of task commands. In task learning, the state of ongoing dialog is used for querying episodic memory during retrospective causal analysis of instructions.

- *How can task goals, structure, and execution knowledge be learned interactively?* The challenge of learning new task definitions online has recently been identified as a challenge problem for integrated intelligent agents (J. Laird, 2014). A variety of approaches have been developed to address this. Prior work (Chernova and Thomaz, 2014) on learning from demonstration, dialog, and reinforcement has addressed the interactive task learning problem in parts. Several initiatives have focused on acquisition of control policies from either human generated embodied traces or reward. However, few if any have studied learning comprehensive representations of tasks from scratch. This thesis takes important steps towards this challenge.

We proposed an interactive variation of explanation-based generalization in Chapter 8 that incrementally learns the proposed task representation. The paradigm is comprehensive it learns availability conditions, task goal definition and recognition, along with learning a hierarchical policy. It satisfies several criteria identified for SII. It learns quickly from sparse data, is able to transfer knowledge across structurally similar tasks, and integrates agent-driven exploration with instructional information. We also demonstrated that the proposed learning paradigm is useful in learning a various functional and organizational tasks. Earlier work on interactive learning does not address these criteria.

Apart from proposing answers to these questions, this thesis also studies the properties of situated interactive instruction and suggests functional characteristics desirable in agents that can use SII to learn new tasks. The desiderata identified here are incomplete and miss some important aspects of communicative agents such as generating the most useful question or response. However, they serve as design goals for interactive learners. We expect that our characterization of the problem generates discussion about what properties are critical to the design of interactive task learners and how they can be formulated and implemented.

9.1 Future Work

The concluding sections of Chapters 8, 6, 7 discussed focused directions of future research. Here, we discuss more broad avenues of work.

This thesis has exclusively focused on learning high-level tasks that can be characterized as a sequence of actions taken in pursuit of some goals. Although language is a suitable modality for learning these tasks, it may be inefficient in some cases. In our formulation, completely describing the goal situation may be difficult for the human instructors as it requires stating several predicates. An easier way is to demonstrate or

label instances of the goal state (Chao et al., 2011) and letting the agent take initiative in generating a goal description and verifying it with the instructor. Another issue is learning primitive control policies. Our formulation assumes a pre-encoded policy and a model for every primitive action. However, there may be situations where the agent has to learn new control policies such as while using a new tool or appliance. The linguistic modality alone may be insufficient to teach these policies and other teaching strategies such as learning from demonstration may have to be employed. With the intention to reduce cognitive load on the instructor, it is worthwhile to investigate methods that allow the instructor to flexibly change the modality of instruction (from explicit instruction to demonstration) or provide mixed-modality instruction (combine demonstration with language to draw attention to relevant features). This will lead to a more comprehensive account of interactive task learning.

A different avenue is learning from reading, which does not involve real-time interaction but still has the advantage of learning in a social construct. There are several resource on the web such as WikiHow that describe how various tasks can be performed. In situations where direct interaction is infeasible, the agent may look up these resources to learn task representations. Learning usable task representations from reading requires substantial advancements in language understanding models. Traditional NLP methods are not suited for agents and recent situated comprehension methods (such as those proposed in this thesis) require the participants to be co-located. The Indexical approach has the potential to assist in learning from reading. Through initial interactive experience, the agent can align its knowledge of the world with natural language. This interactive experience can build common ground that can be used to understand written text. The alignment of words with perceptual aspects, semantic categorization, and knowledge of tasks in the world will be useful in simulating an appropriate situation that provides non-linguistic context to the written text. The agent can explore various ways of performing a task and apply instructions on this simulated situation. This experience, then, can be used

to extract useful task representation. There are several applications where this capability is useful. Most games (*Infinite Mario*, *Civilization*) and equipment (cellphones, computers) are accompanied by manuals containing information about how to play or operate them. Agents can learn useful conceptual and procedural knowledge from reading manuals and can use this knowledge to play a better game or assist human users in device operation and debugging. This line of research also finds use in easy extension of virtual training software (such as those deployed in armed forces) to incorporate new scenarios by providing their linguistic descriptions.

This thesis has taken an *agent-oriented* view of interactive task learning and proposes representations and methods that let an intelligent agent interact with humans to gather information about new tasks and consequently induce general task representations. The design of our methods is motivated by observations of human behavior in prior work. However, these observations do not capture the entire variation and complexity of human behavior. Therefore, it is essential to evaluate these methods in a human-robot/agent interaction (HR/AI) scenarios in order to develop truly robust and flexible paradigms. There are several questions that may be answered through such studies. Our models have a few parameters (exploration depth for learning, attention strategy, dialog heuristics) that can be fit to what humans expect from their collaborative partners. Other questions include the variability in instructions, preference for specific instruction strategy, and most critically, the errors made in instruction and how those can be remedied by further interactions.

9.2 Conclusion

Our research aims to develop intelligent collaborators that not only interact naturally with their human collaborators but also adapt to novel environments and tasks. This dissertation takes step critical to achieving this goal. Language is the primary modality of

human communication. It is expressive, flexible, and easy to use for humans. It allows humans to establish shared beliefs about the environment and learn from each other's experiences. However, its expressiveness and contextual flexibility poses significant challenges in developing a language faculty for intelligent agents. This thesis studies how exploiting non-linguistic contexts and knowledge of the domain can be useful in resolving some ambiguities pervasive in language. Being able to sustain a linguistic interaction allows an intelligent agent to communicate and learn about its environment. As language can be used to encode a variety of information required for learning new tasks, it can be used to develop powerful task learning paradigms. This thesis presents an example of language-driven learning paradigms - situated interactive instruction - that exploits flexible task-oriented dialog to learn a variety of tasks. The learned task representations not only are useful in producing behavior in the environment but also crucially provide the means through which language (verbs) are grounded in perceptions and domain knowledge. Consequently, learning new tasks also facilitates communicate about them using verbs.

Appendix
Definitions

Identifier. An identifier is a node in the working memory graph.

Working memory element. A working memory element (WME) is a triple of three symbols: an identifier, an attribute, and a value. A template of for a working memory is:
`(indentifier ^attribute value)`
While the `identifier` must be an existing node in the working memory graph, the `attribute` and the `value` may be either terminal constants or non-terminal graph nodes.

Rule. A rule (or a production) has three components: a name, a set of conditions (also called the left-hand side or LHS), and a set actions (also called the right-hand side or RHS). A template of a rule is:

```
sp {production-name
    condition 1
    condition 2
    ...
    -->
    action 1
    action 2
    ...}
```

A condition is a pattern for matching on or more WMEs. Each condition consists of

a test for the identifier, and the tests for augmentations of that identifier - attributes and values. The test in conditions can be a variable that matches against constants in WMEs in identifier, attribute, and value positions. An action adds new WMEs in the working memory.

Long-term Identifier. Long-term identifier (LTI) are identifiers (nodes) that exist in semantic memory graphs. The alpha-numeric string that labels an LTI is permanently associated with that LTI: any retrievals of the LTI are guaranteed to return the associated alpha-numeric label.

Semantic memory **cue**. A semantic memory cue is composed of WMEs that describe the augmentations of an LTI. A cue-based semantic retrieval performs a search for a long-term identifier in semantic memory whose augmentations exactly match an agent-supplied cue, as well as optional cue modifiers.

Episodic memory **cue**. An episodic memory cue is composed of WMEs that partially describe a top-state of working memory in the retrieved episode. Cue-based episodic memory retrieval commands are used to search for an episode in the store that best matches an agent-supplied cue.

Episode. An episode captures the entire top-state of working memory at a time instance.

Further details can be found in the Soar Manual (J. E. Laird et al., 2014).

Bibliography

Allen, J., Chambers, N., Ferguson, G., & Galescu, L. (2007). Plow: A Collaborative Task Learning Agent. In *Proceedings of the National Conference on Artificial Intelligence*. Vancouver, BC.

Apple. (2013). Use your voice to do even more with siri. Retrieved from http://www.apple.com/ios/siri/

Argall, B., Chernova, S., Veloso, M., & Browning, B. (2009). A Survey of Robot Learning from Demonstration. *Robotics and Autonomous Systems*.

Atkeson, C. & Schaal, S. (1997). Robot Learning from Demonstration. In *Proceedings of the International Conference on Machine Learning* (Vol. 97, pp. 12–20).

Bailey, D. R. (1997). *When push comes to shove: a computational model of the role of motor control in the acquisition of action verbs* (Doctoral dissertation, University of California, Berkeley).

Bangalore, S., Di Fabbrizio, G., & Stent, A. (2008). Learning the Structure of Task-Driven Human-Human Dialogs. *IEEE Transactions on Audio, Speech, and Language Processing*, *16*(7), 1249–1259.

Bentivegna, D., Atkeson, C., & Gordon, C. (2004). Learning Tasks from Observation and Practice. *Robotics and Autonomous Systems*, *47*, 163–169.

Brenner, M., Hawes, N., Kelleher, J. D., & Wyatt, J. L. (2007). Mediating Between Qualitative and Quantitative Representations for Task-Orientated Human-Robot Jnteraction. In *Proceedings of the Twentieth Joint International Conference on Artificial Intelligence* (pp. 2072–2077). Hyderabad, India: Morgan Kaufmann.

Byron, D. & Fosler-Lussier, E. (2005). *The OSU Quake 2004 Corpus of Two-Party Situated Problem-Solving Dialogs*. The Ohio State University.

Cakmak, M. & Takayama, L. (2013). Towards a Comprehensive Chore List for Domestic Robots. In *Proceedings of the Eighth International Conference on Human-Robot Interaction*.

Cantrell, R., Scheutz, M., Schermerhorn, P., & Wu, X. (2010). Robust Spoken Instruction Understanding for HRI. In *Proceedings of the Fifth ACM/IEEE International Conference on Human-Robot Interaction* (pp. 275–282).

Cantrell, R., Talamadupula, K., Schermerhorn, P., Benton, J., Kambhampati, S., & Scheutz, M. (2012). Tell Me When and Why to do it! Run-time Planner Model Updates via Natural Language Instruction. In *Proceedings of the Seventh International Conference on Human-Robot Interaction*.

Cassell, J. (2000a). *Embodied Conversational Agents*. MIT press.

Cassell, J. (2000b, April). Embodied conversational interface agents. *Communications of Association of Computing Machinery*, *43*(4), 70–78.

Cassell, J., Bickmore, T., Campbell, L., Chang, K., Vilhjálmsson, H., & Yan, H. (1999). Requirements for an Architecture for Embodied Conversational Characters. In *Computer Animation and Simulation* (pp. 109–120). Springer.

Chai, J., She, L., Fang, R., Ottarson, S., Littley, C., Liu, C., & Hanson, K. (2014). Collaborative Effort Towards Common Ground in Situated Human-robot Dialogue. In *Proceedings of the Ninth ACM/IEEE International Conference on Human-robot Interaction* (pp. 33–40). HRI '14. Bielefeld, Germany: ACM.

Chambers, C. G., Tanenhaus, M. K., & Magnuson, J. S. (2004, May). Actions and Affordances in Syntactic Ambiguity Resolution. *Journal of experimental psychology. Learning, memory, and cognition*, *30*(3), 687–96.

Chao, C., Cakmak, M., & Thomaz, A. (2011). Towards Grounding Concepts for Transfer in Goal Learning from Demonstration. In *Proceedings of the international conference on development and learning*.

Chen, D. & Mooney, R. (2011). Learning to Interpret Natural Language Navigation Instructions from Observations. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence* (pp. 859–865). San Francisco, CA: AAAI Press.

Chen, X., Ji, J., Jiang, J., & Jin, G. (2010). Developing High-Level Cognitive Functions for Service Robots. In *Proceedings of Ninth International Conference on Autonomous Agents and Multiagent Systems* (pp. 989–996).

Chernova, S. & Thomaz, A. (2014). Robot Learning from Human Teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *8*(3), 1–121.

Childers, J. B. & Tomasello, M. (2002). Two-year-olds Learn Novel Nouns, Verbs, and Conventional Actions from Massed or Distributed Exposures. *Developmental psychology*, *38*(6), 967.

Deits, R., Tellex, S., Thaker, P., Simeonov, D., Kollar, T., & Roy, N. (2013). Clarifying Commands with Information-Theoretic Human-Robot Dialog. *Journal of Human-Robot Interaction*, *2*(2), 58–79.

DeJong, G. & Mooney, R. (1986). Explanation-based Learning: An Alternative View. *Machine Learning*.

Dietterich, T. (2000). The MAXQ Method for Hierarchical Reinforcement Learning. *Journal of Artificial Intelligence Research*, *13*, 227–303.

Erol, K., Hendler, J., & Nau, D. (1992). UMCP : A Sound and Complete Procedure for Hierarchical Task-Network Planning. *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*.

Gentner, D. (1982). *Why Nouns are Learned before Verbs: Linguistic Relativity Versus Natural Partitioning*. Center for the Study of Learning, University of Illinois at Urbana Champaign.

Gentner, D. (2006). Why Verbs are Hard to Learn. *Action Meets Word: How Children Learn Verbs*, 544–564.

Glenberg, A. & Robertson, D. A. (1999). Indexical understanding of instructions. *Discourse Processes*, *28*(1), 1–26.

Google. (2013). Google Now. The Right Information at Just the Right Time. Retrieved from http://www.google.com/landing/now/

Gorniak, P. & Roy, D. (2004). Grounded Semantic Composition for Visual Scenes. *Journal of Artificial Intelligence Research*, *21*, 429–470.

Griffith, S., Subramanian, K., Scholz, J., Isbell, C., & Thomaz, A. L. (2013). Policy shaping: integrating human feedback with reinforcement learning. In *Advances in neural information processing systems* (pp. 2625–2633).

Grollman, D. & Jenkins, O. (2010). Can We Learn Finite State Machine Robot Controllers from Interactive Demonstration. *From Motor Learning to Interaction Learning in Robots: Studies in Computational Intelligence*, *264*, 407–430.

Grosz, B. & Kraus, S. (1996). Collaborative Plans for Complex Group Action. *Artificial Intelligence*, *86*(2), 269–357.

Grosz, B. & Sidner, C. (1986, July). Attention, Intentions, and the Structure of Discourse. *Computational Linguistics*, *12*(3), 175–204.

Gundel, J. K., Hedberg, N., Zacharski, R., & Fraser, S. (1993). Cognitive Status and the form of Referring Expressions in Discourse. *Language*, *69*(2), 274–307.

Huffman, S. (1994). *Instructable Autonomous Agents* (Doctoral dissertation, University of Michigan, Ann Arbor).

Huffman, S. & Laird, J. (1995). Flexibly Instructable Agents. *Journal of Artificial Intelligence Research*.

Imai, M., Haryu, E., & Okada, H. (2005). Mapping Novel Nouns and Verbs Onto Dynamic Action Events: Are Verb Meanings Easier to Learn Than Noun Meanings for Japanese Children? *Child Development*, *76*(2), 340–355. doi:10.1111/j.1467-8624.2005.00849_a.x

Jackendoff, R. (1972). *Semantic Interpretation in Generative Grammar*. MIT Press.

Kamide, Y., Scheepers, C., & Altmann, G. T. M. (2003). Integration of Syntactic and Semantic Information in Predictive Processing: Cross-Linguistic Evidence from German and English. *Journal of psycholinguistic research*, *32*(1), 37–55.

Kaschak, M. P. & Glenberg, A. (2000). Constructing Meaning: The Role of Affordances and Grammatical Constructions in Sentence Comprehension. *Journal of Memory and Language*, *43*(3), 508–529.

Kersten, A. W. & Smith, L. B. (2002). Attention to Novel Objects During Verb Learning. *Child Development*, *73*(1), 93–109.

Kirk, J. & Laird, J. (2014). Interactive Task Learning for Simple Games. *Advances in Cognitive Systems*, *3*, 11–28.

Knoeferle, P. & Crocker, M. W. (2006, May). The Coordinated Interplay of Scene, Utterance, and World Knowledge: Evidence from Eye Tracking. *Cognitive Science*, *30*(3), 481–529. doi:10.1207/s15516709cog0000_65

Knox, W. B. & Stone, P. (2010). Combining Manual Feedback with Subsequent MDP Reward Signals for Reinforcement Learning. In *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems*.

Knox, W. B. & Stone, P. (2012). Reinforcement learning from simultaneous human and mdp reward. In *Proceedings of the 11th international conference on autonomous agents and multiagent systems-volume 1* (pp. 475–482).

Kollar, T., Tellex, S., Roy, D., & Roy, N. (2010). Toward Understanding Natural Language Directions. In *Proceeding of the Fifth ACM/IEEE International Conference on Human-robot Interaction* (pp. 259–267). Osaka, Japan: IEEE Xplore.

Kollar, T., Tellex, S., Roy, D., & Roy, N. (2014). Grounding Verbs of Motion in Natural Language Commands to Robots. In *Experimental robotics* (pp. 31–47). Springer.

Kruijff, G.-J., Lison, P., Benjamin, T., Jacobsson, H., & Hawes, N. (2007). Incremental, Multi-level Processing for Comprehending Situated Dialogue in Human-Robot Interaction. In *Proceedings of the Symposium on Language and Robots* (Vol. 11, pp. 55–64). Aveiro, Portugal.

Laird, J. (2012). *The Soar Cognitive Architecture*. Cambridge, MA: MIT Press.

Laird, J. (2014). *Report on the NSF-funded Workshop on Taskability (Interactive Task Learning)*.

Laird, J. E., Congdon, C. B., Coulter, K. J., Derbinsky, N., & Xu, J. (2014). The soar user's manual. Retrieved from http://web.eecs.umich.edu/~soar/downloads/Documentation/SoarManual.pdf

Laird, J., Rosenbloom, P., & Newell, A. (1986). Chunking in Soar: The anatomy of a General Learning Mechanism. *Machine learning*, *1*(1), 11–46.

Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., & Jurafsky, D. (2012). Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. *Computational Linguistics*, *39*(4), 885–916.

Levin, B. (1993). *English Verb Classes and Alternations*. University of Chicago Press.

Liang, P., Jordan, M., & Klein, D. (2009). Learning Semantic Correspondences with Less Supervision. In *Proceedings of the Forty-Seventh Annual Meeting of the Association of Computational Linguistics* (pp. 91–99). Singapore.

Litman, D. & Allen, J. (1987). A Plan Recognition Model for Subdialogues in Conversations. *Cognitive Science*, *11*(2), 163–200.

Litman, D. & Allen, J. (1990). Discourse Processing and Commonsense Plans. *Intentions in Communication*, 365–388.

Lochbaum, K. E. (1998, December). A Collaborative Planning Model of Intentional Structure. *Computational Linguistics*, *24*(4), 525–572.

Lonsdale, D., Tustison, C., Parker, C., & Embley, D. (2006). Formulating Queries for Assessing Clinical Trial Eligibility. In *Proceedings of the eleventh international conference on applications of natural language to information systems* (pp 82--93). Klagenfurt, Austria: LNCS.

Maclin, R. & Shavlik, J. W. (1996). Creating Advice-Taking Reinforcement Learners. *Machine Learning*, *22*(1-3), 251–281.

MacMahon, M., Stankiewicz, B., & Kuipers, B. (2006). Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions. In *Proceedings of the Twentieth AAAI Conference on Artificial Intelligence* (pp. 1475–1482).

Matuszek, C., Fitzgerald, N., Zettlemoyer, L., Bo, L., & Fox, D. (2012). A Joint Model of Language and Perception for Grounded Attribute Learning. In *Proceedings of the Twenty Ninth International Conference on Machine Learning* (pp. 1671–1678).

Meriçli, Ç., Klee, S., Paparian, J., & Veloso, M. (2014). An Interactive Approach for Situated Task Specification through Verbal Instructions. In *Proceedings of the Thirteenth International Conference on Autonomous Agents and Multi-Agent Systems*.

Mininger, A. (2014). *Maintaining Object Identity in a Cognitive Architecture through Visual-Spatial Reasoning*. Preliminary Examination Report, University of Michigan, Ann Arbor.

Mitchell, T. M., Keller, R. M., & Kedar-Cabelli, S. T. (1986). Explanation-based Generalization: A Unifying View. *Machine Learning*. doi:10.1007/BF00116250

Mohan, S., Kirk, J., & Laird, J. (2013). A Computational Model for Situated Task Learning with Interactive Instruction. In *Proccedings of the Twelfth International Conference on Cognitive Modeling*.

Mohan, S. & Laird, J. (2014). Learning Goal-Oriented Hierarchical Tasks from Situated Interactive Instruction. In *Proceedings of the Twenty Eighth AAAI Conference on Artificial Intelligence*.

Mohan, S., Mininger, A., Kirk, J., & Laird, J. (2012). Acquiring Grounded Representation of Words with Situtated Interactive Instruction. *Advances in Cognitive Systems*, *2*, 113–130.

Mohan, S., Mininger, A., & Laird, J. (2014). Towards an Indexical Model of Situated Language Comprehension for Cognitive Agents in Physical Worlds. *Advances in Cognitive Systems*, *3*, 163–82.

Mutlu, B., Shiwa, T., Kanda, T., Ishiguro, H., & Hagita, N. (2009). Footing in Human-Robot Conversations: How Robots Might Shape Participant Roles Using Gaze Cues. In *Proceedings of the Fourkth ACM/IEEE international conference on Human robot interaction* (pp. 61–68).

Newell, A. & Simon, H. (1972). *Human Problem Solving*. Prentice-Hall Englewood Cliffs, NJ.

Nicolescu, M. & Mataric, M. (2003). Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice. In *Proceedings of the second international conference on autonomous agents and multi-agent systems* (pp. 241–248).

CMUSphinx. (2014). Retrieved from http://cmusphinx.sourceforge.net/

Oviatt, S. L. & Cohen, P. R. (1991). Discourse Structure and Performance Efficiency in Interactive and Non-Interactive Spoken Modalities. *Computer Speech & Language*, *5*(4), 297–326.

Piantadosi, S. T., Tily, H., & Gibson, E. (2012, March). The Communicative Function of Ambiguity in Language. *Cognition*, *122*(3), 280–91.

Poggi, I., Pelachaud, C., de Rosis, F., Carofiglio, V., & De Carolis, B. (2005). Greta. A Believable Embodied Conversational Agent. In O. Stock & M. Zancanaro (Eds.), *Multimodal Intelligent Information Presentation* (Vol. 27, pp. 3–25). Text, Speech and Language Technology. Springer Netherlands.

Rich, C. & Sidner, C. (1998). COLLAGEN: A Collaboration Manager for Software Interface Agents. *User Modeling and User-Adapted Interaction*, *8*(3-4), 315–350.

Roy, D. (2002). Learning Visually Grounded Words and Syntax for a Scene Description Task. *Computer Speech & Language*, *16*(2), 353–385.

Roy, D., Hsiao, K.-Y., & Mavridis, N. (2003). Conversational Robots: Building Blocks for Grounding Word Meaning. In *Proceedings of the HLT-NAACL 2003 Workshop on Learning Word Meaning from Non-Linguistic Data* (pp. 70–77). Association for Computational Linguistics.

Rybski, P. E., Yoon, K., Stolarz, J., & Veloso, M. (2007). Interactive Robot Task Training through Dialog and Demonstration. In *Proceedings of the Second International Conference on Human-Robot Interaction*.

Sammut, C., Hurst, S., Kedzier, D., & Michie, D. (1992). Learning to Fly. In *Proceedings of the Ninth International Workshop on Machine Learning*.

Scheutz, M., Cantrell, R., & Schermerhorn, P. (2011). Toward Human-Like Task-based Dialogue Processing for HRI. *AI Magazine*.

Scheutz, M., Eberhard, K., & Andronache, V. (2004). A Real-Time Robotic Model of Human Reference Resolution using Visual Constraints. *Connection Science*, *16*(3), 145–167.

Schuler, K. (2005). *VerbNet: A Broad-Coverage Comprehensive Verb Lexicon* (Doctoral dissertation, University of Pennsilvania).

Simon, H. (1991). Cognitive Architectures and Rational Analysis: Comment. In *Architectures for Intelligence: The 22nd Carnegie Mellon Symposium on Cognition* (pp. 25–39).

Siskind, J. M. (2001). Grounding the Lexical Semantics of Verbs in Visual Perception using Force Dynamics and Event Logic. *Journal of Artificial Intelligence Research*, *15*, 31–90.

Swartout, W. R., Gratch, J., Hill Jr, R. W., Hovy, E., Marsella, S., Rickel, J., Traum, D. et al. (2006). Toward Virtual Humans. *AI Magazine*, *27*(2), 96.

Tellex, S., Kollar, T., Dickerson, S., & Walter, M. (2011). Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*. San Francisco, CA: AAAI Press.

Thomaz, A., Hoffman, G., & Breazeal, C. (2006, September). Reinforcement Learning with Human Teachers: Understanding How People Want to Teach Robots. In *The Fifteenth IEEE International Symposium on Robot and Human Interactive Communication* (pp. 352–357).

Winograd, T. (1972). Understanding Natural Language. *Cognitive Psychology*, *3*(1), 1–191.