

**CONTRIBUTION TRACKING:
PARTICIPATING IN TASK-ORIENTED DIALOGUE
UNDER UNCERTAINTY**

BY DAVID DEVAULT

A dissertation submitted to the
Graduate School—New Brunswick
Rutgers, The State University of New Jersey
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
Graduate Program in Computer Science

Written under the direction of

Matthew Stone

and approved by

New Brunswick, New Jersey

October, 2008

© 2008

David DeVault

ALL RIGHTS RESERVED

ABSTRACT OF THE DISSERTATION

Contribution Tracking: Participating in Task-Oriented Dialogue under Uncertainty

by David DeVault

Dissertation Director: Matthew Stone

The contribution of this dissertation is to show how interlocutors in dialogue can reason probabilistically about natural language interpretation, dialogue state (context), and natural language generation in a way that is consistent with three fundamental claims made by mainstream theories of pragmatic reasoning in human-human dialogue:

1. interlocutors track and exploit the evolving context to coordinate their individual contributions;
2. the current context depends on what the previous utterances of both interlocutors have meant (contributed);
3. what a speaker can recognizably mean (contribute) by a specific choice of words depends on the current context.

Mainstream pragmatic theories depend on these assumptions to explain how a speaker can make linguistic choices that the hearer will interpret as intended, but these theories do not lend themselves to straightforward probabilistic reasoning. Engineering approaches to building dialogue systems implement straightforward probabilistic reasoning, but sacrifice one or more (sometimes all) of these fundamental aspects of pragmatic theory in order to do so. This dissertation shows how we can achieve the robustness and data-driven methodology enjoyed by engineering approaches while keeping our interlocutors on a sound theoretical footing, and thereby points the way toward a new class of dialogue systems that are empirically driven, that are robust pragmatic reasoners, and that exhibit human-like sensitivity to the ins and outs of language use in context.

Acknowledgments

I have been fortunate to benefit from the perspective and insights of many excellent researchers and fellow students during my graduate years. My work has improved immensely as a result of my interactions with these colleagues.

To begin, during my time at Rutgers, I accrued a tremendous debt to Matthew Stone. Matthew introduced me to computational linguistics. As my interest in dialogue systems developed, he constantly challenged me to think more abstractly and more deeply about each aspect of dialogue modeling and system design that arose, and about how my work relates to the broader research landscape. Just as importantly, countless hours of more free-ranging conversation with Matthew helped me to gradually understand the joys and frustrations of a life of research, and ultimately to accept them as part of my own life. I truly could not have written this dissertation without him.

I have also benefited throughout my graduate years from detailed discussions and camaraderie with fellow students at Rutgers, including in particular Sam Cumming, Kevan Edwards, Phillip Hankins, Natalia Kariaeva, Anubha Kothari, and Iris Oved. Thanks also to the broader audiences of various presentations of this work in the philosophy and computer science departments at Rutgers.

In the summer of 2003, I was fortunate to receive an internship position at Mitsubishi Electric Research Laboratories (MERL), during which I worked with Charles Rich and Candy Sidner on natural language generation for collaborative agents built using their COLLAGEN framework. Though my time at MERL was relatively short, I learned a great deal from Chuck and Candy, including both theoretical and practical insights that have remained influential in my thinking to this date. This included ways of conceptualizing dialogue partners as fundamentally collaborative, and numerous observations about collaborative language use in context. I also encountered the possibility of a system building strategy that aims simultaneously for technical elegance as well as theoretical justification, wherever possible, for the detailed, day-to-day implementation choices a system builder inevitably faces. My time at MERL made me both a better researcher and a better programmer.

I have also been fortunate to have been able to present and discuss the ideas developed in this dissertation at a variety of conferences and workshops over the past several years. Though I think it is fair to say that controversy was as common as consensus in these discussions, they

have substantially improved both the content and the presentation of the work that appears in this dissertation. Thanks especially to Jens Allwood, Luciana Benotti, Harry Bunt, Susan Brennan, Herb Clark, Paul Cohen, Robin Cooper, Kees van Deemter, Raquel Fernández, Kai von Fintel, Jonathan Ginzburg, Jerry Hobbs, Laura Kertz, Staffan Larsson, Alex Lascarides, David Novick, Hannes Rieser, David Schlangen, Rich Thomason, Will Thompson, David Traum, and Mike White.

Thanks of a different kind to many friends and colleagues, including Trina Altman, Matthew Carroll, Christopher Bruce DeVault, Norma DeVault, Mitchell Edmondson, Phillip Hankins, Iris Oved, Aimee Pierce, Farshad Rezai, Cindy Secrest, numerous undergraduates in Rutgers' Spring 2007 Philosophy of Psychology course, and others, who graciously gave their time to participate in ongoing experiments with COREF. These patient souls endured the agent's many quirks, and provided many helpful comments that guided my thinking about its subsequent development. Thanks especially to Iris Oved for helping to coordinate the COREF evaluation reported here.

During the year in which this dissertation was written, I had the good fortune to hold a position as a Visiting Research Assistant at the USC Institute for Creative Technologies (ICT). The perspective I gained from collaborating on the development of several dialogue systems with David Traum and colleagues has substantially improved my understanding of the state of the art in dialogue systems, and has definitely improved the presentation of this work. David Traum's language group, and ICT more broadly, have provided an outstanding environment in which to explore alternative design choices, as well as a fantastic venue for interaction with many exceptional researchers. I count myself lucky to have had this experience while writing my dissertation. I'd especially like to thank David Traum for his perspective and guidance, as well as fellow ICT colleagues Ron Artstein, Luciana Benotti, Sudeep Gandhe, Antonio Roque, Michael Rushforth, Kenji Sagae, and Nicole Whitman.

Many thanks also to my Ph.D. committee: Michael Littman, Chung-chieh Shan, Matthew Stone, and David Traum. They have provided dozens of helpful and insightful comments and suggestions.

Thanks to the National Science Foundation (NSF) for supporting this research. This dissertation is based upon work supported by the NSF under grant numbers HLC 0308121 and HSD 0624191.

As I have pursued my education, I have been blessed by the limitless love and support of my parents, Bruce and Norma DeVault. In so many ways, I would not be where I am today without them. Thanks also to my dear friend Farzad Rezai, whose thoughts and friendship continue to enrich my life each day. And finally, thanks to Aimee Pierce, who was an enthusiastic audience for all of these ideas as they took shape, and whose companionship comforted me every step of the way.

Table of Contents

Abstract	ii
Acknowledgments	iii
List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 Motivation	4
1.2 Relation to previous publications	7
2 Contributing to two-agent, task-oriented dialogue under uncertainty	9
2.1 Two-agent activities	10
2.2 Task-oriented activities	11
2.3 Background: computational models of two-agent, task-oriented activities	15
2.3.1 Models that optimize task performance	16
2.3.2 Deep coherence models	17
2.3.2.1 Joint intentions, SharedPlan, and shared cooperative activity	17
2.3.2.2 Dialogue as a collaboration	21
2.3.2.3 Incremental common ground models	23
2.4 Motivation	25
2.4.1 Evidence for an intermediate level of reasoning in human cognition	26
2.4.2 Independence of specific genres of social interaction	28
2.4.3 The need for flexible contribution tracking across utterances in dialogue	29
2.4.4 The challenge of implementing mutual attitudes probabilistically	30
2.5 Contributive motor action	33
2.5.1 Public vs. tacit actions	33
2.5.2 Action sequences	35

2.5.3	Definition of contributive motor action	36
2.5.4	Task-oriented intentions	37
2.5.5	Expectations of recognition	38
2.5.6	Relation to deep coherence models	40
2.6	Contributive language use	41
2.6.1	Utterances generate dialogue acts	41
2.6.2	Dialogues are tasks	42
2.6.3	Action sequences and dialogue acts	43
2.6.4	Definition of contributive language use	45
2.6.5	Other language use	49
2.7	Interleaving contributive actions with other actions	50
2.8	Relation to interlocutor mental states	53
2.8.1	Relation to Gricean intentions	53
2.8.2	Relation to speech act theory	55
2.9	Conclusion	57
2.9.1	Limitations, caveats, objections	58
3	Contribution tracking in two-agent, task-oriented dialogue	61
3.1	Contribution tracking	61
3.1.1	Motivation	62
3.1.2	Definition of contribution tracking	66
3.1.3	Comparison to POMDP-based dialogue modeling	68
3.2	Collaborative reference	70
3.2.1	Referring as a collaborative activity	70
3.2.2	COREF's object identification game	74
3.3	The RUBRIC and COREF system architectures	77
3.3.1	Contributions to system architectures for dialogue systems	78
3.3.2	An illustrative COREF subdialogue	80
3.3.3	Control flow	83
3.3.3.1	COREF's control flow	87
3.3.4	Dialogue state	89
3.3.4.1	COREF's dialogue state	89
3.3.4.1.1	COREF's model of agent actions	91

3.3.4.1.2	COREF’s models of dialogue subtasks	92
3.3.4.1.3	COREF’s object identification game as a two-agent dialogue task	94
3.3.5	Sensorium	102
3.3.5.1	COREF’s Sensorium	103
3.3.6	Sensory Event Interpreter	104
3.3.6.1	COREF’s Sensory Event Interpreter	105
3.3.6.1.1	Building the horizon graph	107
3.3.6.1.2	Solving an action’s constraints	110
3.3.6.1.3	Assigning probabilities to interpretations	116
3.3.7	Grammar	116
3.3.7.1	COREF’s Grammar	116
3.3.8	Update Control	117
3.3.8.1	COREF’s Update Control	119
3.3.9	Intention Generator	123
3.3.9.1	COREF’s Intention Generator	125
3.3.9.1.1	Generating a contributive utterance while certain about the context	126
3.3.9.1.2	Generating a contributive utterance under uncertainty about the context	132
3.3.9.1.3	Generating a contributive public motor action	138
3.3.10	Motor Apparatus	140
3.3.10.1	COREF’s Motor Apparatus	140
3.4	Conclusion	140
4	Empirical results	142
4.1	Method	142
4.2	Overview of results	143
4.2.1	Collaborative reference while certain about the context	148
4.2.2	Collaborative reference under uncertainty about the context	153
4.2.3	Effect of agent’s role on task performance	165
4.3	New qualitative capabilities	167
4.3.1	Capturing the ambiguity of acknowledgments	167

4.3.2	Implementing ambiguity management as a flexible collaborative activity . . .	174
4.4	New quantitative capabilities	189
4.4.1	Quantifying performance in collaborative ambiguity management	189
4.4.2	Leveraging contribution tracking to tune an agent’s interpretation model . .	193
4.4.2.1	General approach	195
4.4.2.2	Training	197
4.4.2.3	Results	200
4.5	Conclusion	204
5	Conclusion and limitations	205
5.1	Summary	205
5.2	Limitations and future work	206
5.2.1	Other kinds of uncertainty	206
5.2.2	Other clarification and grounding phenomena	207
5.2.3	Reducing the use of hand-crafted domain-specific models	208
5.2.4	Optimizing agent performance	208
5.2.5	Learning to speak collaboratively	209
A	COREF task instructions	210
	Bibliography	214
	Curriculum Vita	221

List of Tables

4.1	Overall distribution of object outcomes.	143
4.2	Number of possible contexts perceived when utterances or actions occur.	143

List of Figures

2.1	A bookcase assembly task.	12
2.2	A possible model of the bookcase assembly task of Figure 2.1.	13
2.3	Central concepts in the joint intentions model of (Cohen and Levesque, 1991).	18
2.4	An example bookcase assembly scenario. Agent <i>A</i> is seated in front of the partially assembled bookcase, which is currently in state s_1 (see Figure 2.1 on page 12). Agent <i>B</i> is standing behind <i>A</i> , with a partially occluded view, due to <i>A</i> 's position.	34
2.5	A simple model of bookcase assembly as a global dialogue task.	44
3.1	A human user plays an object identification game with COREF. The figure shows the perspective of the user (denoted a9). The user is playing the role of director, and trying to identify the square at upper left (indicated to the user by the blue arrow) to COREF. COREF's perspective at the same point in time is shown in Figure 3.2.	75
3.2	COREF plays an object identification game with a human user. The figure shows the perspective of COREF (denoted Agent). COREF is playing the role of matcher, and trying to determine which object the user wants COREF to identify. The user's perspective at the same point in time is shown in Figure 3.1.	76
3.3	An illustrative COREF subdialogue.	81
3.4	The flow of control in a RUBRIC agent.	84
3.5	The flow of control in the COREF agent.	88
3.6	An example COREF dialogue state: s2106.	90
3.7	COREF's CollabRef (D,M,T) task. In this task, the director D helps the matcher M identify a target T as visual object R	93
3.8	Definition of COREF's CreateParallelScenes() task. In this task, two agents collaboratively refer to a sequence of target objects. After each collaborative reference subtask, the matcher adds the object to their scene, and the director clicks continue.	95
3.9	Compiled version of COREF's CreateParallelScenes() task.	96

3.10 COREF's <code>ManageAmbiguity(Observer,SE)</code> task. In this task, <code>Observer</code> has the opportunity to take several actions in response to a sensory event <code>SE</code> in which <code>Observer</code> perceived an action by <code>Actor</code>	97
3.11 COREF's <code>WhichQ(Asker,Knower,AnswerAction)</code> task. In this task, <code>Asker</code> asks <code>Knower</code> a which question.	98
3.12 COREF's <code>YNQ(Asker,Knower,YesAction,NoAction)</code> task. In this task, <code>Asker</code> asks <code>Knower</code> a yes/no question.	99
3.13 COREF's <code>Remind(Asker,Knower,Happened,NoAction,Action)</code> task. In this task, <code>Asker</code> asks <code>Knower</code> whether <code>Knower</code> has done an action, possibly as an indirect request or reminder to perform the action.	100
3.14 The Sensorium module.	102
3.15 The Sensory Event Interpreter module.	104
3.16 The task stack in state s2106.	107
3.17 Horizon graph for s1: <i>ok</i>	107
3.18 A lexical entry for <i>ok</i>	111
3.19 A second lexical entry for <i>ok</i>	112
3.20 COREF's lexical entry for <i>yes</i>	113
3.21 Horizon graph for state s2479.	114
3.22 Horizon graph for state s2465.	115
3.23 The Update Control module.	118
3.24 COREF's Update Control module.	120
3.25 The Intention Generator module.	124
3.26 Possible horizon graph for COREF: <i>the blue circle</i>	126
3.27 Acceptable horizon graph for COREF: <i>the blue circle</i>	128
3.28 SPUD generates <i>the blue circle</i> for COREF. In this figure, each search node considered by SPUD is depicted using only the surface text at that search node. Nodes are ranked vertically, with higher ranked nodes above lower ranked nodes. An underscore indicates the presence of a syntactic gap in the provisional syntactic structure at the node.	131
3.29 Acceptable horizon graph for COREF: <i>did you add it?</i>	133
3.30 Possible horizon graph from state s2209 for COREF: <i>did you add it?</i>	135
3.31 Possible horizon graph from state s2199 for COREF: <i>did you add it?</i>	136
3.32 Acceptable horizon graph for COREF: <code>clickContinue</code>	139

3.33	The Motor Apparatus	140
4.1	Length of object subdialogues.	144
4.2	Mean uncertainty during object subdialogues.	145
4.3	Mean uncertainty vs. <i>correct</i> object outcome during the object subdialogue. There are 435 objects (75.0%) in the <i>correct</i> bin, and 145 objects (25.0%) in the <i>not correct</i> bin.	146
4.4	Mean uncertainty vs. object outcome during the object subdialogue. There are 435 objects (75.0%) in the <i>correct</i> bin, 83 objects (14.3%) in the <i>no object</i> bin, 43 objects (7.4%) in the <i>skipped</i> bin, and 19 objects (3.3%) in the <i>wrong object</i> bin.	147
4.5	Object outcome vs. mean uncertainty during the object subdialogue. There are 516 objects (88.97%) in the leftmost bin, 37 objects (6.38%) in the middle bin, and 27 objects (4.66%) in the rightmost bin.	148
4.6	COREF achieves a <i>correct</i> outcome with a single thread of interpretation	149
4.7	COREF achieves a <i>correct</i> outcome with a single thread of interpretation	151
4.8	COREF achieves a <i>correct</i> outcome after clarifying the user's meaning	154
4.9	COREF achieves a <i>skipped</i> outcome while pursuing several threads of interpretation	159
4.10	Object outcome vs. role assignment. There are 323 objects (55.7%) in the <i>COREF is director</i> bin, and 257 objects (44.3%) in the <i>COREF is matcher</i> bin.	165
4.11	Mean uncertainty vs. role assignment. There are 323 objects (55.7%) in the <i>COREF is director</i> bin, and 257 objects (44.3%) in the <i>COREF is matcher</i> bin.	166
4.12	An ambiguous acknowledgment by subject S14 in a human-human dialogue.	168
4.13	COREF disambiguates an ambiguous acknowledgment	169
4.14	COREF proceeds despite an ambiguous acknowledgment	172
4.15	All questions asked by COREF in the user study	177
4.16	SPUD generates <i>do you mean dark brown?</i> for COREF. In this figure, each search node considered by SPUD is depicted using only the surface text at that search node. Nodes are ranked vertically, with higher ranked nodes above lower ranked nodes. An underscore indicates the presence of a syntactic gap in the provisional syntactic structure at the node.	180
4.17	COREF's clarification question eliminates a perceived ambiguity	181
4.18	COREF's uncertainty is resolved flexibly	185
4.19	COREF's uncertainty is resolved flexibly	187

4.20	Effect of ambiguity management questions on COREF’s uncertainty. At utterance 0, COREF faces an ambiguous context. At utterance 1, COREF has asked a question. Typically, at utterance 2, the user has answered COREF’s question.	190
4.21	Distribution of AMQ effects for COREF’s AMQs.	191
4.22	Relation between object outcome and AMQ effect. This figure shows the outcome that followed each AMQ asked by COREF. These outcomes are grouped according to the effect of the relevant AMQ. The leftmost bin, for AMQ effect -2, represents 4 AMQs. The bin for effect -1 represents 12 AMQs. The bin for effect 0 represents 28 AMQs. The bin for effect 1 includes 1 AMQ.	192
4.23	Distribution in number of interpretations hypothesized from correct states.	197
4.24	The interpretation features, $features(i_{t,j})$, in our learned model.	198
4.25	The observation features, $features(o)$, in our learned model.	198
4.26	The dialogue state features, $features(s_k)$, in our learned model.	199
4.27	Comparison of learned and hand-built models for 2- and 3-way ambiguities.	201
4.28	Comparison of learned and hand-built models for 4- and 5-way ambiguities.	202
A.1	COREF task instructions: page 1 of 3.	211
A.2	COREF task instructions: page 2 of 3.	212
A.3	COREF task instructions: page 3 of 3.	213

Chapter 1

Introduction

This dissertation is part of a project to develop a detailed and robust computational model of natural language dialogue that clarifies and reconciles the linguistic and collaborative reasoning that interlocutors need to perform in conversation. The project aims both for theoretical insights into human pragmatic reasoning as well as an improved practical methodology for building dialogue systems that understand and cooperate with their human interlocutors better.

The contribution of this dissertation is to show how interlocutors in dialogue can reason probabilistically about natural language interpretation, dialogue state (context), and natural language generation in a way that is consistent with three fundamental claims made by mainstream theories of pragmatic reasoning in human-human dialogue:

1. interlocutors track and exploit the evolving context to coordinate their individual contributions;
2. the current context depends on what the previous utterances of both interlocutors have meant (contributed);
3. what a speaker can recognizably mean (contribute) by a specific choice of words depends on the current context.

Mainstream pragmatic theories depend on these assumptions to explain how a speaker can make linguistic choices that the hearer will interpret as intended, but these theories do not lend themselves to straightforward probabilistic reasoning. Engineering approaches to building dialogue systems implement straightforward probabilistic reasoning, but sacrifice one or more (sometimes all) of these fundamental aspects of pragmatic theory in order to do so. This dissertation shows how we can achieve the robustness and data-driven methodology enjoyed by engineering approaches while keeping our interlocutors on a sound theoretical footing, and thereby points the way toward a new class of dialogue systems that are empirically driven, that are robust pragmatic reasoners, and that exhibit human-like sensitivity to the ins and outs of language use in context.

Our approach to achieving this synthesis builds on a conception of dialogue as a series of collaborative contributions by the interlocutors (Clark, 1996). On this view, the individual contributions

that make up a two-person dialogue are realized through action by both parties. First, a speaker selects a signal to express their desired contribution, and then an addressee responds with their own signal to indicate successful recognition of the contribution. Work in this tradition relies on detailed assumptions about the alignment of the interlocutors' mental states, in the form of mutual beliefs or other mutual attitudes, in order to explain how a speaker can make linguistic choices that will allow their contribution to be successfully recognized (Stalnaker, 1978; Clark and Marshall, 1981). Further, the product of a successful contribution is conceptualized as a specific modification of the two interlocutors' common ground of mutual attitudes (Clark, 1996).

In this dissertation, we present a detailed computational model that is largely consistent with this collaborative view of dialogue, but which organizes and explains an interlocutor's collaborative reasoning a little differently. We introduce the notion of *contribution tracking* as a process by which an interlocutor may sometimes be willing to proceed without achieving mutual agreement about which contributions have been made, even when (eventually) identifying the specific contribution is important to the task at hand. During these periods of transient uncertainty, we show that an interlocutor may nevertheless be able to keep talking collaboratively, by using their partial understanding of previous contributions to make linguistic choices that allow them to develop a reasonable expectation that their follow-on contributions will be recognized acceptably.

Importantly, while it can be challenging to assign probabilities to mutual attitudes (DeVault and Stone, 2006), the reasoning involved in contribution tracking is explicitly probabilistic. This offers a number of practical and methodological advantages for building dialogue agents that will inevitably face real-world uncertainties. As this dissertation will show, it enables a theoretically coherent, intuitive, and methodologically workable notion of an agent's uncertainty about what the current context is. It allows system builders to explore more flexible and robust strategies to manage their collaborative agents' uncertainty in interpreting utterances. And finally, by providing a framework within which collaborative agents can use their own dialogue experience to reduce their uncertainty about which contributions their users are trying to make, it makes it possible to start thinking about ways we can build agents that learn to speak collaboratively by talking to their users. In sum, we will argue, contribution tracking significantly clarifies the methodological target of collaborative language use in real-world dialogue systems.

We begin in Chapter 2 by developing a definition of *contributive action* in task-oriented, two-agent dialogue. Our definitions characterize an action (or utterance) as *contributive* just in case it is intended to advance the task, and the actor (or speaker) has a reasonable expectation that their task partner will understand the action (utterance) as it is intended to be understood. Essentially,

a contributive action is a public attempt to advance a two-agent task in a recognizable way. Our definitions provide explicit room for an actor to exploit probabilistic reasoning in formulating an expectation that their intended contribution will be recognized. We relate our definitions to alternative models of task-oriented collaboration, including models that select actions to optimize task performance as well as models that narrate collaboration in terms of evolving mutual attitudes.

We proceed in Chapter 3 to define contribution tracking as a cognitive process in which an agent acts contributively under uncertainty about which prior contributions have been made. We illustrate contribution tracking using the COREF agent (DeVault and Stone, 2007, 2006; DeVault et al., 2005), an implemented dialogue system that collaboratively identifies visual objects with human users. We show that COREF uses contribution tracking as a computational substrate that allows it to collaboratively refer to visual objects with its users. In particular, COREF tries to keep track of the contributions that have been made previously, and to speak contributively, even when it becomes uncertain about what the previous contributions have been.

Our implementation of contribution tracking in COREF also constitutes a precise computational approach to pragmatic reasoning about *tacit actions* (Thomason et al., 2006; DeVault and Stone, 2006) that can affect both the current state of the task and the details of how subsequent utterances should be formulated and interpreted. In particular, we argue, different domain tasks, such as COREF’s object identification task, exhibit their own idiosyncratic patterns of tacit actions and events which interlocutors recognize and exploit for efficient communication. More broadly, as a modeling tool, tacit actions can serve to capture implicit shifts between the different tasks that speakers take up in conversation. By identifying the tacit actions that would allow a public utterance to be interpreted as making a coherent contribution, COREF is often able to derive domain-specific implicatures both for its users’ utterances and for its own utterances. COREF therefore illustrates that including tacit actions as part of a dialogue agent’s contribution tracking can help the agent to identify rich communicative intentions in uncertain conversational situations.

In Chapter 4, we use a human user study to assess COREF’s use of contribution tracking as it carries out its object identification game. We use this study to present several new qualitative and quantitative capabilities that contribution tracking makes possible. Qualitatively, it allows COREF to use flexible inference to identify contributions over time, to adopt a principled collaborative strategy in formulating clarification questions, and to respond robustly to perceived uncertainty in grounding and clarification subdialogues. Quantitatively, it allows us to describe COREF as acting collaboratively to try to reduce its own uncertainty, and to quantify the agent’s ability to do so. We also provide a proof of concept demonstration that COREF’s own experience in contribution

tracking can be used to improve the probabilistic models it relies on to interpret utterances correctly.

We conclude in Chapter 5 with a discussion of some of the limitations of the work reported in this dissertation, as well as some of the future directions this research suggests.

1.1 Motivation

The notion of contribution tracking that is developed in this dissertation is, in part, a way of relating the contributions that speakers can make with their utterances to their uncertainty about what contributions have been made previously. Most dialogue systems summarize the effects of all the previous utterances or contributions in a single representation of the current dialogue *state* or *context*. In this section, to help motivate our approach to modeling an agent’s uncertainty about what context it is in, we review the roles that context representations play in implemented systems and the research methodologies that support these roles.

Most dialogue systems try to solve a number of basic problems. These include recognizing users’ communicative intentions correctly (natural language understanding), selecting an appropriate system response (dialogue policy), and formulating acceptable system utterances (natural language generation). At the level of multi-utterance subdialogues, we would like for the pattern of responses our systems give to be coherent (dialogue modeling). We also generally have design goals at the level of entire dialogues. For example, in task-oriented dialogue, we aim for our systems to have sufficient conversational competence to help users complete the task (Sikorski and Allen, 1996; Rich et al., 2001).

While context may be an important factor in all of these problems, getting a context representation “right” is not generally an explicit design goal. Instead, we aim — often implicitly — for context representations to serve several related functions in our systems:

- **PERFORMANCE.** Context representations should support high-quality solutions to the basic dialogue problems: NLU, dialogue policy, NLG, dialogue modeling, and task/system success.
- **THEORETICAL FOUNDATION.** Context representations should relate a system’s behavior to a theory of (boundedly) rational communication.

Further, as system builders, we are keenly sensitive to the immense scale of the representations and reasoning that seem to be at play in human conversational competence. We thus have the additional desideratum:

- DATA-BASED TECHNIQUES. Context representations should comport with powerful computational methods that tune system representations and behavior to fit real-world data.

From this perspective, we can describe the dialogue systems community as currently split roughly into two camps: an “engineering” camp that aims for PERFORMANCE with context representations determined by DATA-BASED TECHNIQUES, and a “deep coherence” camp that also aims for PERFORMANCE but instead uses context representations that reflect a strong THEORETICAL FOUNDATION.

We include in the engineering camp most systems that characterize dialogue state as following some trajectory through a probabilistically characterized state space such as a MDP (e.g. Levin and Pieraccini, 1997; Levin et al., 1998), POMDP (e.g. Roy et al., 2000; Williams and Young, 2006, 2007; Rieser and Lemon, 2008), or the dynamic Bayesian networks of Horvitz and Paek (2001). This kind of work generally aims to use observed dialogues and task data to learn both the probabilistic parameterization of the state space and a dialogue policy that maximizes a task-based reward signal. A chief motivation for the use of probabilistic state models is the need to overcome the noise that characterizes current automatic speech recognition results. Importantly for our purposes, this work tends to be very practical about what exactly goes into its dialogue state (i.e. context): representations are simply “optimized” to support other design goals (DeVault and Stone, 2006). A good example of the ethos of the engineering camp is Tetreault and Litman (2006), which aims to decide which features to include in a state representation based on the impact those features have on learned dialogue policies.

On the other side of the divide are deep coherence approaches, where researchers aim for good performance using context representations that comport with received theories of rational language interpretation in formal pragmatics (Stalnaker, 1974, 1978; Poesio and Traum, 1997), and with general theories of coordinated activity (Lewis, 1969; Cohen and Levesque, 1991; Grosz and Kraus, 1996) as they apply to dialogue (Clark and Marshall, 1981; Grosz and Sidner, 1990; Lochbaum, 1998). These theories all narrate a successful communicative act in terms of various nested or higher-order beliefs that interacting agents may have about each other. Consequently, complex mental attitudes such as mutual belief or mutual supposition have been seen as a central factor in determining an agent’s context representations (Thomason, 1990; Poesio and Traum, 1997). Dialogue systems research in this tradition includes (Traum, 1994; Matheson et al., 2000; Rich et al., 2001; Allen et al., 2001b; Swartout et al., 2006).

What the dialogue systems community currently lacks, however, is a conceptual framework and practical methodology that allows us to pursue all three goals simultaneously. That is, ideally

we would like to be able to use DATA-BASED TECHNIQUES to develop context representations that support good PERFORMANCE and also confer to the resulting dialogues a strong THEORETICAL FOUNDATION. Developing this framework and methodology is, in a nutshell, the project undertaken in this dissertation.

The basic diagnosis for the present lack of such a methodology is easy to state. The problem is that there is a substantial methodological tension between the standard THEORETICAL FOUNDATION and the use of DATA-BASED TECHNIQUES. The tension has two main sources. The first is that the complex aspects of user and system mental state that theoretical models of context depend on are, in practice, methodologically inaccessible (DeVault and Stone, 2006). When we are building a system, we often simply do not know whether some relevant bit of information – for example, that a certain task step has been completed, or that a certain individual is under discussion – is (or will be) mutually believed or mutually supposed between the system and its human user. The result is that deep coherence models of context cannot draw on a large data set of “ground truth” about what the context was, or should be, at each point to optimize their representation schemes. Instead, context representations must be hand tailored to support intuitive system behavior in a way that seems consistent with a theoretically sound formalism for coordinated communication in dialogue.

The second source of tension is that DATA-BASED TECHNIQUES tend to be most naturally realized in probabilistic models, but the standard THEORETICAL FOUNDATION, which links context to mutual beliefs between interlocutors, is hard to reconcile with the idea that a dialogue agent faces uncertainty about what context it is in (DeVault and Stone, 2006). The end result is that, with respect to their context models, deep coherence approaches tend to sacrifice DATA-BASED TECHNIQUES to secure a THEORETICAL FOUNDATION.¹

Conversely, the pragmatic attitude that the engineering camp takes toward representations of dialogue state can effectively amount to an abandonment of any clear THEORETICAL FOUNDATION for dialogue in favor of straightforward DATA-BASED TECHNIQUES. For example, some researchers pursuing robust human-machine dialogue have found it practical to simply identify the overall dialogue state with the *user's* private state or goal (e.g. Roy et al., 2000; Horvitz and Paek, 2001; Williams and Young, 2007). While this enables coherent, data-based probabilistic reasoning, it abandons the theoretical role of context as a resource by which interlocutors with differing private perspectives coordinate on the assignment of interpretations to utterances (Stalnaker, 1974, 1978).

This dissertation resolves this tension by showing that, contrary to the traditional theoretical

¹Of course, dialogue systems that take a deep coherence approach to their system's context representations may still employ data-based methods to solve other dialogue problems such as NLU, NLG, etc. An example is the dialogue architecture of (Swartout et al., 2006).

arguments, a secure THEORETICAL FOUNDATION does *not* require that a system’s context representations directly track complex aspects of the interlocutors’ mental states. Instead, we develop a new THEORETICAL FOUNDATION, based on a refined model of collaboration in dialogue, which accommodates agents’ real-world uncertainty about prior contributions — and hence, about context. The new framework is shown to be viable by examining the design and PERFORMANCE of an implemented dialogue agent, COREF, and through a proof of concept demonstration that DATA-BASED TECHNIQUES can be used to improve the probabilistic models that COREF’s contribution tracking depends on.

1.2 Relation to previous publications

The content of this dissertation overlaps that of several previous publications. The first publication about COREF was DeVault et al. (2005), which presents a brief survey of how COREF’s design approaches collaborative reference in terms of updates to an evolving information state.

Our methodological critique of mutual attitudes as the organizing principle of context representations began in DeVault and Stone (2006). We expand and clarify this critique in Chapter 2 of this dissertation.

In Thomason et al. (2006), we present a computational model of context update for communicative intentions involving tacit actions. The model of *enlightened update* developed there incorporates the presence of tacit actions into an agent’s pragmatic reasoning, but maintains a theoretical commitment to an *incremental common ground model* in which the context after each utterance is understood in terms of mutual attitudes; see Section 2.3.2.3, below. Chapters 2 and 3 of this dissertation present a largely parallel treatment of pragmatic inference about tacit actions, but here we change the setting of this reasoning by developing the theoretical concepts needed to see an agent as acting contributively without narrating its decision-making in terms of incremental common ground. We start to develop these new concepts in Section 2.5.

The COREF agent we discuss in this dissertation differs from the agents described in DeVault et al. (2005), DeVault and Stone (2006), and Thomason et al. (2006) in that its design now fully reflects our new theoretical view of contribution tracking under uncertainty. In particular, the agent is no longer required to maintain incremental common ground with its users. Instead, COREF now sometimes becomes uncertain about which of several contexts it is in.

The current agent is described in DeVault and Stone (2007). In comparison with that paper, this dissertation includes a substantially expanded discussion of why we conceptualize COREF as

a collaborative user of language. In particular, we crystallize out the new conceptual vocabulary of *contributive action* and *contribution tracking*. This dissertation also presents COREF's pragmatic reasoning in more detail and with more examples, and includes a more thorough discussion of the empirical results reported there. The work reported in Chapter 4 on using machine learning to improve COREF's probabilistic models is new here.

Finally, DeVault et al. (2006) explores the conditions under which a dialogue agent like COREF could be said to speak meaningfully about objects and properties, as a function of the agent's experience and its relation to its human users. We do not explore this theme further in this dissertation.

Chapter 2

Contributing to two-agent, task-oriented dialogue under uncertainty

The spectrum of social activities that agents engage in can be analyzed along various dimensions. For example, (Clark, 1996, ch. 2) characterizes what he calls *joint activities* in terms of the degrees to which they are scripted vs. unscripted, formal vs. informal, verbal vs. nonverbal, cooperative vs. competitive, and egalitarian vs. autocratic, among other factors.

We cannot hope, in the present work, to provide computational models of all the various kinds of social activities that humans and artificial agents might wish to engage in. This dissertation attempts the more limited task of modeling what we shall call *two-agent, task-oriented* activities. We shall be particularly interested in such activities when they lead to a dialogue in which the participants interleave natural language utterances, addressed to their task partners, with other kinds of non-verbal contributions to their task.

In this chapter, we develop a theoretical view of dialogue as a two-agent, task-oriented activity in which agents try to make contributions under uncertainty. We use this theoretical view to define a *contributive action* as a particular kind of public attempt to make a contribution to the task. We show that our characterization of contributive action makes assumptions about an agent's reasoning that are not generally present in performance optimization approaches to task-oriented dialogue, because it gives a special status to being understood by your partner. On the other hand, our model makes weaker assumptions about an agent's reasoning than are required in more theoretical models that explain collaboration in terms of evolving mutual attitudes. We situate the reasoning involved in contributive action as complementary to the kinds of reasoning that are involved in these existing approaches to collaboration.

We begin by defining what we mean by the terms *two-agent* and *task-oriented* in Sections 2.1 and 2.2, respectively. In Section 2.3, we review several existing approaches to building computational models of these kinds of activities. Section 2.4 motivates our new approach in relation to the existing approaches. In Section 2.5, we develop our definition of *contributive action*, with a focus on non-verbal motor actions. Section 2.6 extends this definition to verbal action in order to define

contributive language use in a two-agent task. Section 2.7 discusses how contributive actions may be interleaved with other kinds of actions in real-world tasks. In Section 2.8, we relate our model of contributive language use to Gricean intentions and speech acts — two alternative theories that connect language use more directly to interlocutor mental states. We conclude and discuss several limitations and possible objections in Section 2.9.

Throughout this chapter, we focus on the problem-solving a would-be contributive actor faces at a specific point in a dialogue. In particular, we characterize this problem-solving as one of selecting a contributive action to perform. In the next chapter, we expand our perspective to include an uncertain agent’s reasoning across multiple turns in dialogue. There, we treat contributive action as a conceptual core in a reasoning process of *contribution tracking* that allows interlocutors to act contributively despite transient uncertainties that may persist across several dialogue turns.

2.1 Two-agent activities

We shall use the term *two-agent activity* to refer to activities in which *exactly* two agents participate. Participation will be understood to include three broad types of contributions agents can make to activities. These are verbal contributions (utterances), non-verbal motor actions, and various kinds of purely mental contributions (attention, decision-making, and other kinds of cognitive contributions).

We will describe all these kinds of contributions as *actions*; thus our framework includes purely mental actions as well as more traditional motor actions. The distinction between actions that are publicly observable (such as utterances) and those that are not (such as purely mental contributions, or motor actions that one agent cannot perceive) will play an important role in our model of contributions to task-oriented dialogue.

We limit our scope to activities with no more than two participants to avoid some of the additional complexities that arise in modeling the different roles agents can play in multi-party activities. For example, in multi-party dialogue, third parties may serve as additional addressees, side participants, bystanders, or eavesdroppers (Clark, 1992). It would be interesting to investigate how the framework developed in this dissertation might be adapted to multi-party activities, but we leave this question to future work.

We limit our scope to activities with at least two participants because we are primarily interested in coordination between agents. Thus any implications of this dissertation for modeling (apparently) single-agent language use such as monologue or the composition of written text are indirect, and will not be discussed further here.

To summarize, we give a few examples to illustrate our notion of two-agent activities.

- Activity 1. Agents *A* and *B* move a piano together.
- Activity 2. Agents *A* and *B* paint a house together; *A* paints the bedrooms, and *B* paints all the other rooms.
- Activity 3. Agents *A* and *B*, old friends who have agreed to spend more time together, meet at a café to do so.

2.2 Task-oriented activities

A substantial portion of dialogue systems research has focused on language use during *task-oriented* activities (Allen et al., 2000, 2001a). This research has targeted a range of practical activities in which agents work together to complete some relatively clear and concrete task such as ordering merchandise, accessing information (travel, restaurants, weather, etc.), selecting furniture for a room, or other kinds of relatively simple problem-solving. The motivation for this research focus is twofold. First, there is industrial demand for dialogue systems that can help human users efficiently solve such practical tasks. Second, there is a hunch that developing computational techniques for these kinds of practical tasks is a reasonable approach to making progress on the apparently harder problem of replicating human-level dialogue competence in general (Allen et al., 2000). We share this hunch, and therefore draw on a model of two-agent tasks to develop our general characterization of contributive action under uncertainty in dialogue. We present our particular model of two-agent tasks in this section.

It is possible to take a relatively narrow view of what constitutes a task-oriented activity. For example, one might consider task-oriented activities to include only those in which two agents cooperate toward the achievement of a single high-level goal; see (Perrault et al., 1978), for example. By contrast, we shall adopt a relatively broad view of what makes a two-agent activity task-oriented. At an intuitive level, we aim to require only that there be something the agents are *trying to do* by *taking actions that change the state of their activity*. To keep our framework as general as possible, we shall try to assume very little about the logical structure of what the agents are trying to do, or about how agents represent their task internally. For example, we allow that what the agents are trying to do might be modeled as a single action, as in `move(piano)` for Activity 1. Or it might be modeled as taking a sequence of actions to achieve a specific goal state, as in `isPainted(house)` for Activity 2.

To achieve this flexible view of tasks, we introduce a simple formalism in which a task takes

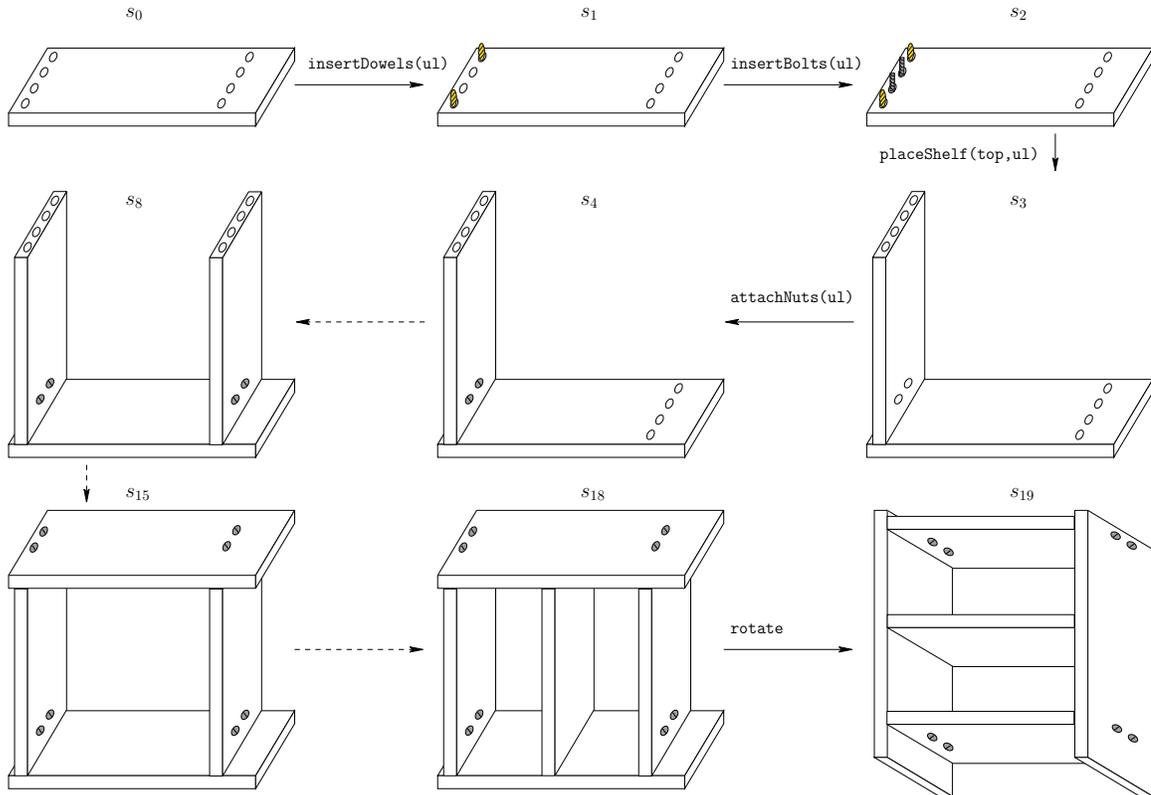


Figure 2.1: A bookcase assembly task.

the form $T = \langle S, I, R, A, N, U \rangle$. S is a set of possible task states. S may have a finite or infinite cardinality. Some state $I \in S$ is the designated initial state. R is a set of “roles” for the participants; one agent will play each role, so $|R| = 2$.¹ A is the set of actions that agents can take. The function $N : R \times S \rightarrow \mathcal{P}(A)$ captures the alternative next actions (the agent in) each role could “coherently” take from a given task state. (N is for **n**ext.) A partial function $U : R \times S \times A \rightarrow S$ captures the state transition or “update” that occurs when the agent playing a given role takes a given action from a given state.²

We will start by illustrating how this model can be applied to a simple two-agent task, and then discuss several subtleties of our approach. We illustrate our model with the simple bookcase assembly task depicted in Figure 2.1. This task is a simplified version of the assembly procedure for the BILLY bookcase sold by IKEA³ circa 2005. In this task, the bookcase is incrementally assembled

¹It is often convenient to conceptualize tasks in terms of roles rather than agents. However, we will sometimes find it convenient and harmless to conflate agents with the roles they play in our discussions of tasks.

²That the update function is partial allows some incoherent or “illegal” combinations of states and actions to leave the task in an undefined state.

³<http://en.wikipedia.org/wiki/IKEA>

$$\begin{aligned}
S &= \{s_0, s_1, s_2, \dots, s_n\} \\
I &= s_0 \\
R &= \{A, B\} \\
A &= \{\text{insertDowels(Loc)}, \text{insertBolts(Loc)}, \\
&\quad \text{placeShelf(Shelf, Loc)}, \text{attachNuts(Loc)}, \dots\} \\
N(r, s_0) &= \{\text{insertDowels(ul)}, \text{insertBolts(ul)}, \text{insertDowels(ll)}, \\
&\quad \text{insertBolts(ll)}\} \text{ for } r \in \{A, B\} \\
N(r, s_1) &= \{\text{insertBolts(ul)}\} \text{ for } r \in \{A, B\} \\
N(r, s_{20}) &= \{\text{insertDowels(ul)}\} \text{ for } r \in \{A, B\} \\
&\dots \\
U(r, s_0, \text{insertDowels(ul)}) &= s_1 \text{ for } r \in \{A, B\} \\
U(r, s_1, \text{insertBolts(ul)}) &= s_2 \text{ for } r \in \{A, B\} \\
U(r, s_0, \text{insertBolts(ul)}) &= s_{20} \text{ for } r \in \{A, B\} \\
&\dots
\end{aligned}$$

Figure 2.2: A possible model of the bookcase assembly task of Figure 2.1.

from component parts, including side panels and shelves, that are secured together using wooden dowels, bolts, and nuts. The figure sketches one sequence of actions that results in the successful assembly of the bookcase.

One way we could model this task in our framework is given in Figure 2.2. We have a set $S = \{s_0, \dots, s_n\}$ of task states, where each state corresponds to some configuration of the partially assembled bookcase that can be reached by agent actions starting from the initial state s_0 , depicted in Figure 2.1, in which the left panel lies flat. Some of these states are indicated in Figure 2.1, but in general there will be many more possible states, since there are various other coherent routes to a successfully assembled bookcase. In this model, we simply identify the roles R with the two agents, which we label A and B . The actions include `insertDowels(Loc)` and `insertBolts(Loc)`, in which wooden dowels or bolts are inserted at a parameterized location, e.g. `ul` (upper end of left panel). The “next state function” $N : R \times S \rightarrow \mathcal{P}(A)$ specifies the actions the agents can coherently take next in each state. This model does not place any role-based constraints on who can act next; if an action is a coherent next action from a given state, either agent can take it. Finally, the “update function” $U : R \times S \times A \rightarrow S$ specifies the state that results when a specific agent takes a specific action from a specific state. Note that this model does not make the result state for an action dependent on which agent took the action.

There are a number of subtle issues at play in this approach to modeling tasks. These issues include how task-oriented agents represent their tasks, the purpose or goal of a task, alignment of agent goals in participating in a task, and what it means for an action to be “coherent” in a given

task state. We now turn to our perspective on these important issues.

This formalism places minimal requirements on the internal representations employed by agents that perform tasks, i.e. task-oriented agents. For example, to participate in such a task, an agent might model the underlying task domain and actions with STRIPS-style operators (Fikes and Nilsson, 1971) or with a similar planning model, but that degree of complexity is not required by our definition of a task. For example, it is clear by inspecting Figure 2.2 that an agent could represent this task with a simple look-up table or with a finite state model. In particular, this task formalism does not require that “task-oriented agents” have the kind of productive planning and plan recognition capabilities that have been a part of task-oriented dialogue systems research since Allen and Perrault (1980). (Nor does it exclude agents with these capabilities.)

Also note that this formalization of tasks, as desired, does not require that there be an explicit goal or purpose for the task. We could, for example, require that all “tasks” aim to achieve a propositional goal. But not all tasks are most naturally characterized in terms of propositional goals (e.g., Activity 3 above). So instead, we operationalize the notion of what the agents are trying to do in the function $N : R \times S \rightarrow \mathcal{P}(A)$ that maps roles and states to coherent next actions. This allows us to leave the factors that make a particular next action coherent as a task-specific detail. Simply requiring that there be some set of coherent next actions at each point will turn out to be sufficient for our characterization of contributive action with respect to a task.

An important caveat in thinking about “what the agents are trying to do” in performing a task is that, in any specific interaction, there may of course not be any *one* thing the agents are trying to do: the agents may be trying to do different things (Clark, 1996, ch. 2). Our approach to contribution tracking, which we present in Chapter 3, supports certain kinds of differences in what tasks the agents think they are performing at any given time. For now, note that by associating the coherent set of next actions with the *task* and *state*, rather than with each individual agent’s view of the task and state, our task model sidesteps this important issue (Pollack, 1986; Grosz and Sidner, 1990; Chu-Carroll and Carberry, 1994). For now, we will simply say that an individual agent might try to compute the set of coherent actions for the task and state it believes it is in by invoking a planner on a goal state, by inspecting a look-up table, by exploring a finite state model of the task, or by some other means.

It is important, however, that there be *some* limits to what next actions count as “coherent,” if the activity is to reasonably be viewed as “task-oriented.” For example, consider a two-agent activity, like Activity 3 above, in which *A* and *B* simply aim to “spend time together.” During their time together, they will each take various actions, but there are relatively few constraints on what actions

they can take at each point. As long as time passes and they are together, their two-agent activity is probably on track. Without further details, describing such an activity as “task-oriented” seems counterintuitive.⁴

By comparison, in our bookcase assembly task, the design of the bookcase places substantive constraints on the coherent next actions at each step. For example, suppose the current task state is s_0 in Figure 2.1, and consider the immediate performance of action `placeShelf(top,ul)`. Placing the top shelf without first inserting the dowels and bolts is contrary to the design of the bookcase: the top shelf will not be able to support any weight unless these parts are in place. So taking this action immediately cannot lead to a successful bookcase assembly; the action will have to be “undone” before success is possible. In this sense, performing this action in state s_0 is a mistake, and would not be a “coherent” next action for either agent to take.

The fact that the coherent next actions in bookcase assembly are relatively constrained is, we propose, one of the reasons we are inclined to view bookcase assembly as a *task*.⁵ Our intuition is that the more the structure and purpose of an activity constrains the possible paths to success, the more natural it will be to describe the activity as a “task”. Our formalization of tasks can capture this intuitive gradation in the relative permissiveness of the function N .

More generally, we suspect that there is no precise way to circumscribe all and only those two-agent activities that could defensibly be called “task-oriented”. Henceforth, we shall focus our attention on activities with a relatively intuitive task structure and purpose.

2.3 Background: computational models of two-agent, task-oriented activities

So far, we have developed a characterization of two-agent, task-oriented activities, but we have yet to say what it means for two agents to *work together* or *coordinate* on a task, as opposed to simply *doing* it. Computer scientists and engineers have pursued a number of approaches to building computational models of two-agent, task-oriented activities. In this section, we survey two broad classes of computational models, and discuss the way they approach coordination and the methodologies associated with each.

⁴ A and B might choose to carry out some task-oriented activity as a *way* of spending time together, but this in itself would not make “spending time together” a task-oriented activity.

⁵Compare the range of coherent actions in this related activity: Agents A and B spend time together, in the presence of the parts of an unassembled bookcase. In such an activity, taking action `placeShelf(top,ul)` from state s_0 might well be perfectly coherent.

2.3.1 Models that optimize task performance

One way to approach computational modeling of a two-agent task is to focus on some measure of task performance. In the case of our bookcase assembly task, for example, task performance could be measured in various ways: how long does it take, on average, for the two agents to assemble the bookcase? how many mistakes (missing dowels, missing bolts, shelf placed backwards, etc.) are present in the assembled bookcase, on average? on what percentage of bookcase assembly attempts is the bookcase assembled without any mistakes? how many actions are taken, on average, in a bookcase assembly attempt? Given such a quantitative performance measure, one can try to build a computational model of an agent’s decision-making that tries to optimize the task performance.

The performance optimization approach to task-oriented dialogue is illustrated, for example, by Levin and Pieraccini (1997) and Levin et al. (1998), who write (1997), “a dialog system is a machine that tries to achieve an application goal in an efficient way through a series of interactions with the user.” Levin and Pieraccini model a dialogue as a trajectory through a Markov Decision Process (MDP), which allows them to treat performance optimization as a problem of identifying an optimal policy in a reinforcement learning setting. In their travel information task, where one agent provides requested information to the other, they define performance as a weighted sum of several factors including expected number of turns in the interaction, expected amount of information retrieved and presented to the user, and whether any information is eventually presented to the user.

The performance optimization approach also includes work that models dialogue as a Partially Observable Markov Decision Process (POMDP), using hand-crafted rewards based on task performance measures (e.g. Roy et al., 2000; Williams and Young, 2006, 2007). We discuss the POMDP approach further in Section 3.1.3 (page 68). Another modeling choice is the dynamic Bayesian networks of Horvitz and Paek (2001). In general, work in the performance optimization approach characterizes dialogues and other two-agent, task-oriented interactions as following some trajectory through a probabilistically characterized state space, and aims to use observed dialogues and task data to learn both the probabilistic parameterization of the state space and an agent policy that maximizes a task-based performance measure or reward signal. From our perspective, this approach has three important methodological strengths:

1. the computational models employed are probabilistic and support inference under uncertainty;
2. free parameters in the computational models can be tuned to fit large data sets;
3. the approach supports objective evaluation of dialogue systems.

On the other hand, in systems that optimize task performance directly, it can be difficult to isolate any explicit reasoning that agents use to coordinate their independent contributions to the task. Rather, coordination errors are controlled, to the extent that they affect task performance, and thus can be “optimized out” of the agent’s action policy.

The lack of explicit attention to coordination is not intrinsic to performance optimization approaches, however. For example, one source of coordination errors is that one agent misunderstands which action another agent has taken. Horvitz and Paek (2001) use a probabilistic dialogue model to decide, following a user request about which the system is uncertain, whether requesting that the user repeat the request, or another system action, would maximize expected utility. If utility tracks task performance, this can be viewed as a performance optimization approach to coordination.

More generally, optimization approaches can be directly applied to achieving good coordination in task-oriented interaction, but, to date, this is the exception rather than the rule. In tasks where coordination is crucial to performance, a common approach is to hand-author domain-specific rules (action policies), where needed, rather than rely on a general theory of coordination; see Tambe (1997). We now turn to another class of models in which an agent’s reasoning is fundamentally organized around an explicit theory of coordination.

2.3.2 Deep coherence models

In this section, we review several models that aim to achieve what we might call “deep coherence” in an agent’s reasoning about coordination in non-verbal tasks as well as in task-oriented dialogue. We begin with a discussion of several influential theoretical models that can be applied to various kinds of tasks. We then discuss a view of dialogue in particular as a collaboration between the interlocutors. We conclude with a discussion of the incremental common ground model, which is a way of modeling coordination between interlocutors in dialogue.

2.3.2.1 Joint intentions, SharedPlan, and shared cooperative activity

System builders who attempt to implement a general theory of coordination often rely on elements of several influential theoretical models that were developed beginning in the early 1990s. These are the SharedPlan model (Grosz and Sidner, 1990; Grosz and Kraus, 1996; Lochbaum, 1998), the *joint intentions* model (Levesque et al., 1990; Cohen and Levesque, 1991), and the *shared cooperative activity* model (Bratman, 1992). In this section, we survey these theoretical models, emphasizing the role that agent mental states play in the theories.

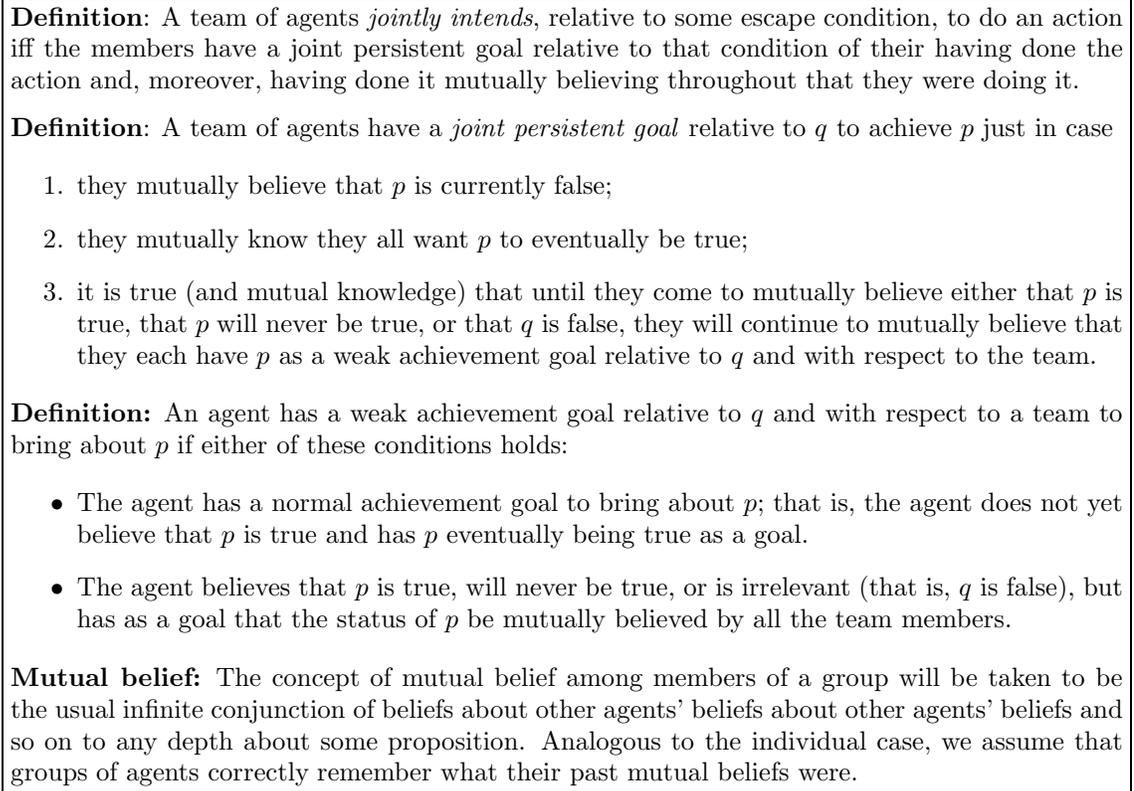


Figure 2.3: Central concepts in the joint intentions model of (Cohen and Levesque, 1991).

In general, these models each introduce a number of technical concepts to try to capture some of our intuitions about how agent mental states (beliefs, goals, intentions, etc.) evolve (or should evolve) as two or more agents work together on a task. For example, (Cohen and Levesque, 1991) introduces the concepts of *joint intention*, *joint persistent goal*, and *weak achievement goal* to characterize how a multi-agent team can work together to perform an action or achieve some proposition p . These concepts are defined in Figure 2.3. The intuitive concept of teamwork between agents is analyzed as the presence of a *joint intention*, which reduces to various conditions on agent mental states. For example, the agents must mutually believe that p is not already achieved. The agents must also mutually know that they all want p to be true. Further, condition 3 in the definition of *joint persistent goal* requires that the team continue to mutually believe that all the individual agents either have p as a goal or else have the goal of conveying why p is no longer goal-worthy to the other team members. Intuitively, this keeps the team “united” in its belief that the individual agents will remain engaged in the team’s effort to bring p about. The joint intentions model has remained influential in subsequent work on explicitly collaborative implemented systems; see e.g. Jennings

(1995), Tambe (1997), and Nair and Tambe (2005).

The definitions of Figure 2.3 serve to illustrate some important features that are common to all three models (*SharedPlan*, *joint intention*, and *shared cooperative activity*). First, these models attempt to characterize with remarkable precision how the mental states of the various agents are related as they work together on the task. For example, in Figure 2.3, the technical notions of *mutual belief* and *mutual knowledge* each effectively ascribe an *infinite* set of beliefs to *each* agent in the team at *each* point in the task. To see this, consider (2.1), which is one of the first and most widely known definitions of mutual belief (Schiffer, 1972):

$$\begin{aligned}
 & \mathbf{B}_A p \\
 & \wedge \mathbf{B}_B p \\
 & \wedge \mathbf{B}_A \mathbf{B}_B p \\
 \mathbf{MB}_{A,B} p =_{\text{def}} & \wedge \mathbf{B}_B \mathbf{B}_A p \\
 & \wedge \mathbf{B}_A \mathbf{B}_B \mathbf{B}_A p \\
 & \wedge \mathbf{B}_B \mathbf{B}_A \mathbf{B}_B p \\
 & \dots
 \end{aligned} \tag{2.1}$$

The definition records an infinite, hierarchical interrelation between the private beliefs of agents A and B about some proposition p . The modal operators \mathbf{B}_A and \mathbf{B}_B represent the beliefs of A and B , respectively. The requirement of mutual belief makes the presence of teamwork dependent on a remarkable number of fine-grained and detailed beliefs; for example, whether agent A believes agent B believes agent A believes agent B believes that the goal p is already attained.

The basic rationale for requiring such mutual attitudes is that when mutual attitudes are not guaranteed, coordination errors are possible (Lewis, 1969; Clark and Marshall, 1981). For example, imagine that agents A and B both have as a goal that exactly one agent move into some grid location (x, y) . Suppose however that A doesn't believe B has this goal, and nor does B believe A has this goal. In this case, both A and B might move into grid location (x, y) . Thus neither agent achieves their common goal because discrepancies in their belief states lead to a coordination failure. To some extent, we would be disinclined to describe their behavior as "coordinated".

The standard line of reasoning (Clark and Marshall, 1981) concludes that agents should ensure that they have mutual attitudes in order to avoid such coordination failures. Because multi-agent task-oriented activities are generally ripe with potential for coordination errors, requirements that agents have mutual attitudes pervade all three of these theoretical models of multi-agent coordi-

nation. For example, the *joint intentions* model in Figure 2.3 places six separate requirements for mutual belief or mutual knowledge on the team members. Similarly, SharedPlan (Grosz and Kraus, 1996) frames collaborative planning in terms of mutual beliefs, while *shared cooperative activity* (Bratman, 1992) requires the stronger condition of common knowledge (Lewis, 1969).

A second commonality among these three models is an attempt to respond in very precise ways to specific human intuitions about detailed collaborative scenarios. For example, Cohen and Levesque (1991), in discussing their definition of a *joint persistent goal*, write:

Thus, if a team is jointly committed to achieving p , they mutually believed initially that they each have p as an achievement goal. However, as time passes, the team members cannot conclude about each other that they still have p as an achievement goal, but only that they have it as a *weak* achievement goal; each member allows that any other member may have discovered privately that the goal is finished (true, impossible, or irrelevant) and be in the process of making that known to the team as a whole. If at some point, it is no longer mutually believed that everyone still has the normal achievement goal, then the condition for a joint persistent goal no longer holds, even though a mutual belief in a weak achievement goal will continue to persist. This is as it should be: if some team member privately believes that p is impossible, even though the team members continue to share certain beliefs and goals, we would not want to say that the team is still committed to achieving p .

This means, for example, that if a group of seven soccer players is attempting to score via some maneuver x which they have practiced many times, but one of the seven has decided it is impossible to score with that maneuver and is in the process of signaling this judgment to the others, then we can no longer describe these seven soccer players as *jointly intending* to score with maneuver x . Suppose the other six players are too busy to notice the skeptical player's signal, and somehow manage to go ahead and score with maneuver x . (The skeptical player's further participation turned out not to be crucial.) Can we say that the seven players' *joint intention* succeeded? The definitions say no: the seven players did not technically have a *joint intention* at the time the success occurred. Perhaps the definitions support the claim that the remaining six players had a *joint intention* that succeeded, but this is somewhat unclear, since those six players thought their "team" comprised seven players rather than six.

The point we want to emphasize here is that these models of coordinated interactions are very subtle in their focus and implications, and consequently they formalize slightly different types of social activities. For example, Bratman (1992)'s model of *shared cooperative activity* specifically excludes any kind of coercion among the cooperating agents; further it must be common knowledge among all the agents that no coercion is afoot. So a team that does have a *joint intention* may not be engaged in a *shared cooperative activity* due to an element of coercion, or due merely to a suspicion of coercion that prevents it being common knowledge that there is no coercion. Similarly,

agents that have a SharedPlan may not have a *joint intention*, since the existence of a SharedPlan depends only on mutual belief among the collaborating agents, while *joint intention* further requires mutual knowledge of certain things.

2.3.2.2 Dialogue as a collaboration

This dissertation aims to advance the view that natural language dialogue between human speakers is usefully viewed as a kind of collaborative activity. This perspective on dialogue is perhaps most explicit in the work of Herb Clark and colleagues (e.g. Clark and Marshall, 1981; Clark and Wilkes-Gibbs, 1986; Clark and Schaefer, 1989; Clark, 1996). Proponents of this perspective are motivated by several lines of research that suggest that the linguistic choices human speakers frequently make in dialogue have much in common with the decision-making that agents generally need to perform to collaborate on more concrete tasks. More specifically, when examined closely, human-human dialogues seem to have a detailed structure in which numerous subtasks are temporarily taken up by the interlocutors. Further, detailed investigation of the behavior of interlocutors during these subtasks suggests that the interlocutors themselves approach these subtasks as miniature collaborative projects which they work together with their dialogue partners to solve.

For example, frequently in dialogue, a speaker needs to identify to their hearer some object from their shared environment. Linguistically, this often shows up through the use of a referring expression such as “the conference table”, “the long table”, or simply “the table”. Because there are generally many alternative referring expressions that might suffice, a speaker has one or more linguistic choices to make in selecting a particular expression to use on a particular occasion. While a speaker *could* view these choices as unilateral and irrevocable, it turns out that, in practice, reference frequently plays out as a multi-utterance process in which both the original speaker and hearer actively participate (Clark and Wilkes-Gibbs, 1986). For example, in the following dialogue, Clark and Wilkes-Gibbs (1986) identify “Monday” as an other-corrected noun phrase:

- B* : How long y’gonna be here?
A : Uh- not too long. Uh just til uh Monday.
B : Til- oh yih mean like a week f’m tomorrow.
A : Yah
B : (continues)

Sometimes the hearer not only clarifies, corrects, or amends the speaker’s original noun phrase,

but actually *supplies* the noun phrase for the speaker. For example, Clark and Wilkes-Gibbs (1986) classify “uh” in the following dialogue as a proxy noun phrase, which is used to indicate that the speaker faces a problem in generating an appropriate referring expression:

A: The tree has, uh, uh...

B: Tentworms.

A: Yeah.

B: Yeah.

As discussed further in Section 3.2.1 (page 70), Clark and Wilkes-Gibbs (1986) identify a number of phenomena of this sort, in which successful reference involves the active participation of both the initiator of the reference and the audience for the reference. When taken together, these phenomena suggest that reference in dialogue looks very much like a collaborative subtask in which two interlocutors temporarily work together toward the goal of coming to an agreement about the identity of an object that one of them intends to single out to the other.

In addition to reference, there are several other kinds of evidence that argue for a collaborative view of some of the subtasks involved in natural language dialogue. For example, when interlocutors refer repeatedly to the same object or property in dialogue, over time, they tend to eventually settle on the same terms, i.e. to make the same lexical choices, to do so (Brennan and Clark, 1996). For example, if *A* and *B* are talking at length about a certain long table, it is generally unlikely that *A* will repeatedly refer to it as “the conference table” while *B* persists in referring to it as “the long table” (unless there is a specific reason for their different lexical choices). Rather, their descriptions will tend to converge, for example to “the long table” (Brennan and Clark, 1996). This phenomenon of *lexical entrainment* seems to simplify the interpretation process for the hearer of a referring expression. Intuitively, the use of agreed lexical items such as “long” and “table” enables more rapid interpretation, with less risk of misinterpretation. The fact that, in human-human dialogue, both interlocutors participate in lexical entrainment of each other’s terms suggests that they view their lexical choices in a collaborative setting in which they are each trying to rapidly and accurately single out specific objects or properties to each other.

A more explicitly collaborative set of phenomena in human-human dialogue fall under the heading of *grounding* (Clark, 1996, ch. 8). The frequent occurrence of grounding utterances like *right*, *ok*, *what did you say?*, *did you say **Barack** will win?*, and many others, suggests that human interlocutors do a significant amount of explicit, interactive work to make sure they understand each

other’s utterances as a dialogue progresses. It is natural to view grounding utterances as part of a collaboration in which the interlocutors work together to achieve the goal of (sufficient) mutual understanding about what has previously transpired in their dialogue (Clark and Schaefer, 1989). Grounding will be discussed further in Section 2.3.2.3 (page 23) and in Chapter 4.

While many dialogue system builders view language use in dialogue as collaborative in various ways, most dialogue systems do not implement a formal theory of collaboration or coordination such as SharedPlan (Grosz and Sidner, 1990; Grosz and Kraus, 1996; Lochbaum, 1998), *joint intentions* (Levesque et al., 1990; Cohen and Levesque, 1991), or *shared cooperative activity* (Bratman, 1992), as discussed in Section 2.3.2. The main exception we are aware of is COLLAGEN (Rich et al., 2001), which implements Lochbaum (1998)’s approach to interpreting discourse using the SharedPlan framework. COLLAGEN agents aim to identify the hierarchical structure in a discourse, and to associate discourse segments (Grosz and Sidner, 1986) with shared goals between the system and user. COLLAGEN agents have generally accepted input in the form of menu selections which are then interpreted using recipe-based plan recognition as contributions to solving open shared goals. An agenda determines how the system itself should work towards solving open goals. Once a system contribution is selected, a natural language utterance may be generated either through a template-based system or by invoking a grammar-based generator that exploits the current discourse structure to formulate a recognizable utterance (DeVault et al., 2004).

The COREF agent described in this dissertation is similar to a COLLAGEN agent in many ways, especially in the definition of hierarchical task models for collaboration, the use of these task models to interpret actions and utterances as specific contributions to ongoing tasks, and the commitment to a theoretical view of dialogue as fundamentally collaborative. COREF differs from a COLLAGEN agent, however, in that its architecture (presented in Chapter 3) is designed to support pervasive probabilistic reasoning. This includes probabilistic reasoning about natural language understanding, dialogue state, and natural language generation. This dissertation can be viewed, to a large extent, as building on the insights embodied in COLLAGEN, but telling a new story that shows how an agent can assimilate these kinds of representations and reasoning into a probabilistic framework.

2.3.2.3 Incremental common ground models

Most dialogue systems in the deep coherence tradition (see e.g. Traum, 1994; Matheson et al., 2000; Allen et al., 2001b; Swartout et al., 2006) employ what we will call an *incremental common ground representation* to try to maintain coordination between the system and user. In this approach, systems try to represent the *common ground* (Clark, 1996) they have with the user, but they do not

necessarily exploit a detailed formal model of coordination or collaboration to guide their linguistic decision-making. With each new utterance in the dialogue, these systems attempt to establish what new information has become common ground as a result of the utterance: this is the *incremental* part.

As a simple example, suppose it is a point of contention between Larry and Jason that Larry always travels to meet Jason, but Jason never travels to meet Larry. Imagine the following dialogue fragment as Larry relays the day’s events to his friend Jeff:

Larry₁: You’ll never believe it. Jason came to my place today!

Jeff₂: Wow, ok.

According to the incremental common ground model, both Larry and Jeff have to decide, after utterance Larry₁, whether the new content of Larry₁ has become common ground between Larry and Jeff. In this example, it seems to have, as evidenced by Jeff’s response “Wow, ok.” at Jeff₂. However, actually establishing what new information has become common ground may lead to additional utterances, by one or both interlocutors, in a process known as *grounding* (Traum, 1994; Clark, 1996, ch. 8). For example, this dialogue might have unfolded as follows:

Larry₁: You’ll never believe it. Jason came to my place today!

(2.2) Jeff₂: Your office or your house?

Larry₃: My office.

Jeff₄: Wow, ok.

In this variation, after Larry₁, Jeff is uncertain what the intended referent of Larry’s referring expression “my place” was. Jeff voices this uncertainty by asking “Your office or your house?” at Jeff₂. On an incremental common ground model, the intended referent of “my place” in Larry₁ seems to not really become common ground until after Larry₃, or perhaps not until after Jeff₄.

Systems built in the incremental common ground framework try to explicitly represent, as the dialogue unfolds, what content from each new utterance has become common ground between the system and the user. The rationale that underlies this approach is twofold. First, common ground is a theoretical notion that either comprises or entails the existence of mutual attitudes between interlocutors (see Clark, 1996, p. 93–100). Therefore, in theory, if interlocutors track their common ground correctly, they can eliminate the possibility of coordination problems in their use of language (Clark and Marshall, 1981). This general dedication to avoiding miscoordination captures one broad aspect of human-human collaboration in dialogue: that human speakers generally want to be un-

derstood correctly by their hearers, and often seem to engage in additional reasoning and careful lexical choice to avoid misunderstanding. (However, note that tracking common ground alone does not guarantee finer-grained collaboration on subtasks such as reference and lexical choice.)

A second motivation is that the incremental common ground approach affords an intuitive theoretical narration of grounding and clarification utterances such as *right, ok, your office or your house?, what did you say?, did you say **Barack** will win?*, and many others. The theoretical story is that all these utterances, and the subdialogues in which they occur, are attempts by interlocutors to manage their private representations of the common ground. In other words, these various kinds of “grounding” utterances seem to indicate that human interlocutors themselves employ an incremental common ground representation as a dialogue unfolds.

We discuss some of the strengths and weaknesses of the incremental common ground model in the next section.

2.4 Motivation

In this dissertation, we develop, implement, and evaluate a new theoretical model of how interlocutors can coordinate their contributions to task-oriented dialogue. We view this model as articulating an independent level of reasoning that is consistent with both performance optimization and deep coherence approaches to dialogue. This new level of reasoning is primarily concerned with how agents who face real-world uncertainty can try to make their own contributions to task-oriented dialogue, and recognize the contributions of others, as the dialogue unfolds over time. We call this reasoning CONTRIBUTION TRACKING.

We view contribution tracking as an independent level of reasoning because it can be implemented, as we will demonstrate, independently of either task optimization or deep coherence modeling. On the other hand, we believe that the inference involved in contribution tracking is a useful “cognitive building block” that could be exploited in dialogue systems that include task optimization and/or deep coherence modeling. We therefore situate contribution tracking as one among three broad, non-exclusive approaches to modeling an agent’s reasoning in task-oriented dialogue. In order of increasing sophistication in the reasoning an actor (speaker) performs about their audience’s mental state:

1. task performance optimization
2. contribution tracking

3. deep coherence modeling

The development of contribution tracking as an independent level of reasoning has a number of motivations. We describe these motivations in the remainder of this section.

2.4.1 Evidence for an intermediate level of reasoning in human cognition

There is some evidence from the cognitive psychology of infants for a type of reasoning in human communication that is more sophisticated than simple task optimization, but less sophisticated than the reasoning involved in deep coherence models. In this section, we summarize a few findings that suggest that an intermediate level of reasoning occurs in human infants.

Human infants engage in behavioral turn-taking from their first months of life (Tomasello et al., 2005), and begin to engage in goal-directed, coordinated two-agent activities (with their parents) at around 9 months of age (Carpenter et al., 1998). Between the ages of 9 to 15 months, infants gradually begin to participate in “multi-modal interactions” with adults: they follow adults’ gaze, attend to adults’ attitudes about unfamiliar objects and situations, imitate actions performed by adults on objects, and use their own communicative gestures (pointing or object-showing) to direct adults’ attention (Carpenter et al., 1998; Tomasello et al., 2005). During this same time period, infants begin to interpret the actions of others as goal-directed, to use means-end analysis to predict the actions that others will take, and to decide between alternative plans that can achieve the same goal that others have been observed to achieve (Tomasello et al., 2005; Gergely, 2002).⁶

Infants at 12 months of age will “repair” an adult’s interpretation of a pointing gesture when the adult misunderstands which object the infant is pointing to (Liszkowski et al., 2007). (One way infants can repair the adult’s understanding is by repeating the pointing gesture to the correct object.) 12 month old infants’ repair behavior is sensitive both to the adult’s identification of the correct object and the adult’s positive vs. disinterested attitude toward the object. Liszkowski et al. (2007) interpret this finding as suggesting that “by twelve months of age infant declarative pointing is a full communicative act aimed at sharing with others both attention to a referent and a specific attitude about that referent.”

Infants typically begin using language to communicate at around 13 to 14 months of age (Tomasello et al., 2005). By 18 months, infants are sensitive to whether their verbal requests (e.g. “Ball!”) have been understood correctly, and may repair an adult’s misunderstanding even if their request

⁶Generally, the exact ages at which specific abilities emerge in infants are matters of current research, and are subject to differing interpretations by different psychologists; see (Gergely, 2002) for a recent overview. Our arguments in this section are not sensitive to small deviations from the ages reported here.

is otherwise satisfied (e.g. they are given the ball they requested, but the adult appears to have misunderstood them as wanting another object) (Hauser et al., unpublished).

Importantly for our purposes, the emergence of these abilities to engage in coordinated two-agent activities, including participation in natural language dialogue and repair of misunderstandings by their partners, *precedes* the development of a mature theory of mind in human children (Tomasello and Rakoczy, 2003). Though the exact age at which a theory of mind develops is a matter of interpretation, difficulties in tasks that require attributing false beliefs to others have led many to conclude that children generally do not have a mature theory of mind until around 4 years of age; see (Gergely et al., 1995) for discussion. For our purposes, the important point is that it does not appear plausible that infants exploit a complex theory-of-mind based representation such as mutual belief (see (2.1) on page 19) at ages of 9-15 months, when they begin participating in coordinated, two-agent, task-oriented activities, including dialogue.

This leads us to conclude that reasoning about mutual attitudes is not a necessary condition for (rudimentary) participation with human beings in coordinated, two-agent, task-oriented activities, including dialogue. Rather, when infants begin to participate in dialogue, they seem to exploit a more basic level of reasoning that presumably draws on their existing abilities to point, direct and share attention, interpret others' actions as goal-directed using means-end analysis, entertain alternative plans, assess whether misunderstandings have occurred, and repair misunderstandings when appropriate to communicative goals. Thus, initial participation in dialogue by infants seems to involve richer communicative goals than are typically modeled in task optimization approaches (e.g., being *understood*, not just getting what you want efficiently), but less sophisticated reasoning than is suggested by deep coherence models of dialogue.

We take these findings to suggest that there may be an intermediate level of reasoning about coordination in human cognition. While we do not argue that our specific model of contribution tracking is identical to the reasoning that infants perform at any specific age, we do take the ability of young infants to participate in dialogue as suggestive of the kinds of inference that might profitably be included in implementations of contribution tracking. Further, we view these findings about human development as suggestive that an intermediate level of reasoning in artificial dialogue systems may serve as a useful building block in implementing the more sophisticated reasoning of older children and adults — which may more closely match deep coherence models.

In conclusion, our attitude toward the level of reasoning we articulate in this dissertation is consistent with the perspective of Tomasello et al. (2005) on what aspect of human cognition underwrites the unique social and cultural achievements of our species:

Our proposal for this “small difference that made a big difference” is an adaptation for participating in collaborative activities involving shared intentionality – which requires selection during human evolution for powerful skills of intention reading as well as for a motivation to share psychological states with others. ...

There are two other main theoretical contenders for what makes human cognition unique in the animal kingdom. First, of course, many theorists point to language, and without a doubt language must play a central role in all discussions of the evolution of human cognition. But saying that only humans have language is like saying that only humans build skyscrapers, when the fact is that only humans (among primates) build freestanding shelters at all. Language is not basic; it is derived. It rests on the same underlying cognitive and social skills that lead infants to point to things and show things to other people declaratively and informatively, in a way that other primates do not do, and that lead them to engage in collaborative and joint attentional activities with others of a kind that are also unique among primates. ...

The other major contender for what makes human cognition unique is theory of mind. Our proposal is of course one variant of this, and indeed we would argue that the full understanding of intentional action, including its rational and normative dimensions, involves some understanding of things mental. But when most people use the term theory of mind they mean the belief-desire psychology with which school-age children and adults operate. But this form of theory of mind is clearly derivative of more basic social-cognitive skills. Thus, Tomasello and Rakoczy (2003) argue and present evidence that while the understanding and sharing of intentions emerges ontogenetically in all cultural settings at around 1 year of age – with no known individual differences due to environmental factors – the understanding of beliefs emerges some years later at somewhat different ages in different cultural settings, and there is very good evidence that participating in linguistic communication with other persons (especially some forms of perspective-shifting discourse) is a crucial, perhaps even necessary, condition for its normal development. And so again, while the understanding of beliefs and desires is clearly a critical component in uniquely human cognition and culture, we do not believe it is basic, but rather it, too, is derived from the understanding and sharing of intentions.

One way to view the theory and implementation of contribution tracking set out in this dissertation, then, is as a knowledge level (Chapter 2) and computation level (Chapter 3) hypothesis about the underlying mechanisms involved in the basic ability human beings have to understand and share intentions with each other.

2.4.2 Independence of specific genres of social interaction

A second motivation for the development of contribution tracking as an independent level of reasoning arises from the fine-grained distinctions that deep coherence models tend to apply to social interactions. As discussed in Section 2.3.2.1, models that define rich relationships between an agent’s decision-making and its attitudes toward other agents tend to circumscribe what appear to be subtle “genres” of coordinated behavior. Thus, joint intentions apply to a certain kind of team (characterized by specific kinds of goals, *inter alia*), SharedPlan applies to a certain kind of collaborative planning (characterized by mutual belief about plan steps, *inter alia*), and shared cooperative activity applies to a certain kind of activity (characterized by mutual knowledge and lack of coercion, *inter alia*). Our perspective is that these are all valuable models of the relevant genres of interaction.

1. assume that all that has become common ground is that Jason came to some place or other of Larry’s, or
2. assume that Larry meant Larry’s office, and that it has become common ground that Jason came to Larry’s office, or
3. assume that Larry meant Larry’s house, and that it has become common ground that Jason came to Larry’s house, or
4. ask a clarification question, as in (2.2) on page 24 above.

However, to force Jeff to immediately address the ambiguity, by choosing one of these options, seems unnecessary. For example, suppose Jeff expects Larry to continue this dialogue with further details about the meeting with Jason, and that in all likelihood these additional details will disambiguate the location of the meeting. In this case, Jeff might respond “Wow, ok.” as indicated in Jeff₂. There are various ways the ambiguity might be resolved; here are three examples. Larry might utter Larry₃¹, confirming the office interpretation of “my place”. Or Larry might utter Larry₃², from which Jeff might infer that Larry meant his office, since Larry and Jason never work on scripts at Larry’s house. Or Larry might utter Larry₃³, from which Jeff might infer that Larry meant his office, since Jeff knows Larry was at his office at 3pm. It is easy to image various other ways that Jeff might deduce what Larry originally meant by “my place”, after one or several additional utterances by Larry and Jeff. The disadvantage of the *incremental* common ground approach is that it seems to require that the matter be resolved immediately after Larry₁, while in practice dialogue partners can employ flexible inferential strategies to single out each other’s intended contributions over time.

Our view is that the most flexible approach to this inference is to frame it as probabilistic inference. We discuss the prospects for a probabilistic approach to common ground in the next section.

2.4.4 The challenge of implementing mutual attitudes probabilistically

One way to respond to examples like (2.3) above would be to try to define a probabilistic representation of common ground for use in a dialogue agent’s reasoning. To do so, however, seems to require two developments, each of which seems methodologically difficult.

The first requirement would be to identify a data set or corpus that specifies what the common ground actually *is* at each point in time for a set of dialogues. Given such a corpus, various statistical techniques could be used to build a probabilistic model of common ground for use in

an agent’s reasoning. Unfortunately, it can be very difficult to actually establish what is common ground between two interlocutors at any given point in a dialogue. In fact, if common ground is understood in terms of mutual knowledge, and the communication channel between interlocutors is noisy, establishing what is common ground is impossible even in principle (Halpern and Moses, 1984). If common ground is understood rather in terms of mutual belief or mutual supposition, it can still be quite difficult, methodologically, for a system builder to adjudicate whether the system and its user have (or will have) common ground about any given utterance (DeVault and Stone, 2006). Thus it is difficult to establish a data set of “ground truth” that can serve as the target for a probabilistic model of common ground.

Partly due to these difficulties, implemented systems generally employ a heuristic *grounding criterion* (Clark and Schaefer, 1989) to decide in what circumstances the system will assume that something is common ground. Concretely, this shows up in the presence of hand-built, heuristic context update rules (Larsson and Traum, 2000). For example, such an update rule might predict that it is not until after Jeff’s utterance of “Wow, ok.” at Jeff₄ in (2.2) on page 24 that Larry’s intended referent for “my place” becomes common ground. It is less clear what update rule should apply in a subdialogue like (2.3), where no clarification occurs, but uncertainty about the speaker’s contribution persists.

The second requirement for a useful probabilistic model of common ground would be to develop a theoretically clear interpretation of what a numeric probability that something is common ground *means*, and to relate this interpretation to the agent’s inference in dialogue. Here again, meeting this requirement is difficult if common ground is understood in terms of mutual attitudes (DeVault and Stone, 2006). To see this difficulty, let us assume the relevant attitude is belief, and that mutual belief is defined by (2.1) on page 19, which we repeat here as (2.4) for convenience:

$$\begin{aligned}
 & \mathbf{B}_A p \\
 \wedge & \mathbf{B}_B p \\
 \wedge & \mathbf{B}_A \mathbf{B}_B p \\
 \mathbf{MB}_{A,B} p =_{\text{def}} & \wedge \mathbf{B}_B \mathbf{B}_A p \\
 & \wedge \mathbf{B}_A \mathbf{B}_B \mathbf{B}_A p \\
 & \wedge \mathbf{B}_B \mathbf{B}_A \mathbf{B}_B p \\
 & \dots
 \end{aligned} \tag{2.4}$$

We will consider several approaches to modeling mutual belief probabilistically. We will be particularly interested in how an agent A can determine the probability $P_A(\text{MB}_{A,B} p)$ to assign to p being mutually believed between A and B .

First approach: all belief is bivalent. On this approach, for each proposition p , each agent either believes p or does not believe p . Unfortunately, whether mutual belief holds between A and B for any proposition p is itself a proposition, viz. (2.4). So for each proposition p , each agent either believes that mutual belief obtains for p , or does not believe mutual belief obtains for p . This effectively rules out an agent having substantive uncertainty about mutual belief.

To be explicit, if we assume agents have introspective access to their own beliefs (they know for each proposition whether they believe it or not), then we must have:

$$P_A(\text{MB}_{A,B} p) = \begin{cases} 1 & \text{if } A \text{ believes } p \text{ is mutually believed by } A \text{ and } B \\ 0 & \text{otherwise} \end{cases}$$

Thus, if belief is bivalent, agents cannot have non-trivial uncertainty about what their mutual beliefs with their partners are. The bivalence prevents it.

Second approach: all belief is graded. On this approach, for each proposition p , each agent assigns a degree of belief to p . Let us simply refer to A 's degree of belief in p by $B_A p$, and further assume that degrees of belief are probabilities, so that $P_A(p) = B_A p \in [0, 1]$.

On this approach, since all belief is graded, we are faced with re-defining mutual belief, since the propositional conjuncts in (2.4), and indeed the nested propositions that appear within the conjuncts, have all been replaced with probabilities. For example, when belief was bivalent, we could view a conjunct like $B_A B_B p$ as true iff A (bivalently) believes B (bivalently) believes p . What shall we replace such a conjunct with?

Since all belief is graded, on this approach, our replacement should capture the fact that B 's belief in p is itself probabilistic. For example, suppose $B_B p = 0.4$. We could therefore re-write a conjunct like $B_A B_B p$ as $B_A(B_B p = 0.4)$, and use this expression to refer to A 's degree of belief that B 's degree of belief in p is 0.4. Similarly, we could rewrite $B_B B_A B_B p$ by something like $B_B(B_A(B_B p = 0.4) = 0.0081)$. However, clearly we are rapidly moving away from a methodologically tractable notion of a dialogue system's uncertainty. If it is difficult to identify a data set or corpus of ground truth about bivalent mutual belief, it seems dramatically more difficult to identify the ground truth about the specific "nested degrees of belief" that may obtain in particular situations.

Another strategy is that of Monderer and Samet (1989). Their formalism replaces the bivalent belief of a single agent with the graded notion of p -belief in proposition C , or belief that the probability of C is *at least* p . Building on this notion, they give a set theoretic definition of *common p -belief* in a proposition C , which holds whenever some event occurs such that all agents assign a probability of *at least* p to C . Common p -belief has hierarchical properties that can be related to the conjuncts of (2.4). There are additional formal models of mutual attitudes that might be amenable to some form of probabilistic treatment; see Vanderschraaf and Sillari (2007) for a survey of alternative formalisms. While it might be possible to implement inference in dialogue over a notion like common p -belief, or another probabilistic version of mutual attitudes, we are unaware of anyone who has attempted to implement such a notion in an actual system, or articulated how such an approach could be motivated by empirical data.

Though we do not presently see how mutual attitudes can be implemented probabilistically, we do not take this as a reason not to exploit, whenever possible, an estimate of the mutual attitudes that hold between a system and user. On the contrary, where mutual attitudes can be determined, they are clearly helpful in assessing the soundness of an agent’s decision-making. Rather, our view is that there may be periods of *transient uncertainty* in dialogue during which a different type of reasoning is more appropriate. When uncertainty can be adequately resolved, the results of this reasoning might even inform updated estimates of mutual attitudes. This is, therefore, another motivation for our definition of contribution tracking as an independent level of reasoning in a dialogue agent.

2.5 Contributive motor action

In this section, we draw on the model of tasks set out in Section 2.2 to define contributive action as a kind of public attempt to make a recognizable contribution to a two-agent, task-oriented activity. Our focus in this section is non-verbal motor action. We delay our discussion of language use to Section 2.6.

2.5.1 Public vs. tacit actions

Recall from the discussion in Section 2.1 that we allow purely mental actions such as decision-making steps as explicit actions in our tasks. Additionally, we allow non-verbal motor actions (e.g. pressing a button, picking something up, etc.) that, when taken by A , are not observable by B . We call these purely mental actions and any (token) motor actions that are not observable *tacit actions*. We call other actions *public actions*.

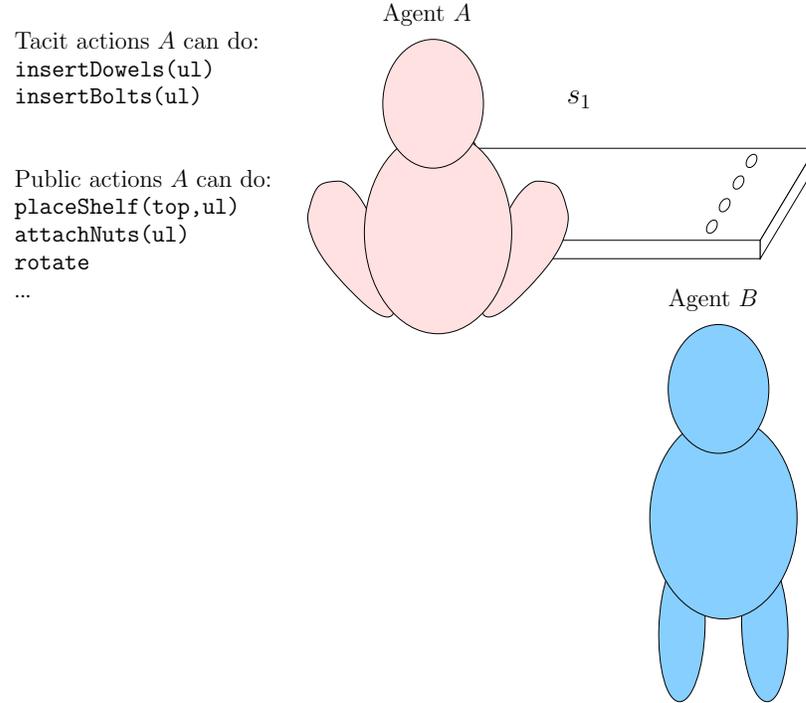


Figure 2.4: An example bookcase assembly scenario. Agent A is seated in front of the partially assembled bookcase, which is currently in state s_1 (see Figure 2.1 on page 12). Agent B is standing behind A , with a partially occluded view, due to A 's position.

We assume that whenever A takes any public action, B observes that A has taken *some* action, but we allow that B may face perceptual uncertainty about exactly *which* action A has taken. By contrast, we assume that B does not observe anything at all when A takes a tacit action.⁸

For illustration purposes, let us stipulate some additional details in a particular bookcase assembly task undertaken by agents A and B . (To review the bookcase assembly task, see Figure 2.1 on page 12 and Figure 2.2 on page 13.) Figure 2.4 shows a scenario in which agents A and B might find themselves. Agent A is seated in front of the partially assembled bookcase. Agent B is standing behind A , with a partially occluded view, due to A 's position. Let us suppose that the occlusion and other details of B 's perceptual field make it the case that if A takes action `insertDowels(ul)` or `insertBolts(ul)`, B will not perceive that A has taken any action. B will however perceive other actions by A , including `placeShelf(top,ul)`.

⁸In Thomason et al. (2006), we counted an action as tacit if some of the collaborators do not know whether it has taken place *or whether it has had the appropriate effects*. In this dissertation, we do not encounter any cases in which an action is observed but the observing agent is uncertain about whether the action's intended effects have been achieved. Therefore, we adopt a simpler model in which an action's effects are assured whenever the action is taken, and no action is viewed as tacit solely because the observing agent is uncertain about the actual effects of the action they have observed. Any action that is observed is public.

In such a scenario, we say that A 's actions `insertDowels(ul)` and `insertBolts(ul)` are *tacit*, while A 's other actions are *public*. The importance of this distinction for contribution tracking is that, when there are both tacit actions and public actions in a task, agents by necessity can only expect to coordinate using the evidence provided by their *public* actions. To capture this necessity, we will use a function `public($X, \langle a_1, \dots, a_n \rangle$)` which, given an action sequence $\langle a_1, \dots, a_n \rangle$ taken by agent X , returns the sequence of zero or more actions in $\langle a_1, \dots, a_n \rangle$ that are public, preserving their order. Thus, for example, `public($A, \langle \text{insertBolts(ul)}, \text{placeShelf(top, ul)} \rangle) = \langle \text{placeShelf(top, ul)} \rangle$` . We will use this function to capture the public evidence a sequence of actions by one agent provides to the other agent.

2.5.2 Action sequences

According to the model of tasks we adopted in Section 2.2, in any given task state s , there is a set $N(A, s)$ of actions that agent A could coherently perform next. We have not assumed that tasks have a strict turn-taking structure, so in general agent A may coherently perform multiple actions in succession without any intervening action by B . Due to this, and especially due to the presence of tacit mental and motor events, we will define contributions in terms of action *sequences* rather than single actions.

We shall use the following notation for sequences. We use angle brackets to denote a sequence of actions in the chronological order of their execution, e.g. $\langle a_1, \dots, a_n \rangle$.⁹ We use a parenthetical ordered tuple notation to represent the concatenation of two or more sequences: $(\langle a_1, \dots, a_n \rangle, \langle b_1, \dots, b_m \rangle) = \langle a_1, \dots, a_n, b_1, \dots, b_m \rangle$, $(\langle a_1, \dots, a_n \rangle, \langle b_1, \dots, b_m \rangle, \langle c_1, \dots, c_k \rangle) = \langle a_1, \dots, a_n, b_1, \dots, b_m, c_1, \dots, c_k \rangle$, etc. We will generally mark sequence-valued variables and functions with asterisks, e.g. $a^* = \langle a_1, \dots, a_n \rangle$.

For convenience, we define a recursive function U^* that captures the state that results if agent A carries out an action sequence a^* starting from state s :

$$U^*(A, s, a^*) = \begin{cases} s & \text{if } a^* = \langle \rangle \\ U^*(A, U(A, s, H), T) & \text{if } a^* = (\langle H \rangle, T) \text{ \& } U(A, s, H) \text{ is defined} \\ \text{undefined} & \text{otherwise} \end{cases} .$$

For example, we can formalize a few of the state transitions illustrated in Figure 2.1 on page 12

⁹Note that we thus assume a task domain in which actions do not overlap temporally. This is not generally the case in spoken dialogue, where simultaneous speech sometimes occurs rather than strict turn-taking (Allen et al., 2001b). Nevertheless, in practice, strict turn-taking can be assumed or enforced in many dialogue systems, such as COREF.

as follows:

$$\begin{aligned}
U^*(A, s_0, \langle \text{insertDowels}(\text{ul}) \rangle) &= s_1 \\
U^*(A, s_0, \langle \text{insertDowels}(\text{ul}), \text{insertBolts}(\text{ul}) \rangle) &= s_2 \cdot \\
U^*(A, s_0, \langle \text{insertDowels}(\text{ul}), \text{insertBolts}(\text{ul}), \text{placeShelf}(\text{top}, \text{ul}) \rangle) &= s_3
\end{aligned}$$

Using this update function U^* , we can define the set of coherent non-empty action sequences, $N^*(A, s)$, that A can perform next starting from task state s :¹⁰

$$N^*(A, s) = \{\langle a_1, \dots, a_n \rangle \mid a_1 \in N(A, s) \wedge \dots \wedge a_n \in N(A, U^*(A, s, \langle a_1, \dots, a_{n-1} \rangle))\}.$$

For example, in our bookcase assembly task, we have:

$$N^*(A, s_1) = \{\langle \text{insertBolts}(\text{ul}) \rangle, \langle \text{insertBolts}(\text{ul}), \text{placeShelf}(\text{top}, \text{ul}) \rangle, \dots\}.$$

2.5.3 Definition of contributive motor action

With this notation in place, we now define how A can act contributively in a task-oriented activity with B in several steps.

- **Definition:** Agent A has a *task-oriented intention* to do action sequence a^* iff A intends to do a^* as part of a plan to advance the task, i.e.:
 1. in the (probabilistic) belief that the current task state is some state s , and
 2. in the (probabilistic) belief that $a^* \in N^*(A, s)$.
- **Definition:** Agent A has an *expectation of recognition* for action sequence a^* iff A has a rationale according to which A expects (probabilistically) that B will infer from observing $\text{public}(A, a^*)$ that
 1. A has (probably) done a^* , and
 2. A has (probably) done a^* in an attempt to advance the task.
- **Definition:** Agent A does a^* contributively iff
 1. A has a task-oriented intention to do a^* ,

¹⁰This definition assumes that $\forall s [a \in N(A, s) \rightarrow U(A, s, a)$ is defined]. I.e., updates are always defined for coherent next actions.

2. A has an expectation of recognition for a^* , and
3. A does a^* .

We'll begin with a few comments about the high-level structure of these definitions, before turning to a more detailed discussion of task-oriented intentions and expectations of recognition.

The first thing to note is that we have placed only two substantive conditions on the classification of an actor as acting contributively. One requires that the action(s) be intended by A to make a contribution to the task. This therefore limits our definition to task-oriented activities. The other requires that A have some rationale for believing that B will recognize the specific action(s) that A has performed and that A performed them in an attempt to contribute to the task.

The second thing to note is that we explicitly allow A to exploit probabilistic reasoning in formulating its task-oriented intention and in developing its expectation of recognition. Our assumption of pervasive probabilistic reasoning in coordinating agents is in sharp contrast with the assumption of pervasive mutual attitudes in the deep coherence models reviewed in Section 2.3.2.1. Where those models attempt to completely eliminate the risk of coordination errors through the presence of mutual attitudes, our model expects coordinating agents to estimate the probability that possible contributions are coherent and recognizable, and to use these estimates to try to choose contributions that are reasonably believed to be coherent and recognizable.

2.5.4 Task-oriented intentions

Our model depends on the notion of a *task-oriented intention*. We understand an intention as a relation between an agent and a plan: specifically, a relation in which the agent intends to perform the actions in the plan, in order to carry out the plan (Pollack, 1990). Our representation of intentions distills the agent's plan down to an assumed task state and a sequence of actions that are motivated by the plan. In a certain sense, this representation captures the "what", but only partially captures the "why", in an agent's plan to act. We view this simplification as a convenient abstraction that glosses over some of the detailed planning inference that many agents likely perform in order to choose actions that are enabled by specific assumptions about task state, and that bring about specific desired effects (Stone, 2004).

As an example of a task-oriented intention, suppose A and B are assembling a bookcase in the scenario of Figure 2.4 on page 34. Suppose A 's mental state includes the following assessments:

$$\begin{aligned}
P(\text{state} = s_1) &= 0.9942 \\
P(\langle \text{insertBolts}(\text{ul}), \text{placeShelf}(\text{top}, \text{ul}) \rangle \in N^*(A, s_1)) &= 0.8650.
\end{aligned}
\tag{2.5}$$

Partly on the basis of these assessments, suppose A adopts the intention to carry out the action sequence:

$$s^* = \langle \text{insertBolts}(\text{ul}), \text{placeShelf}(\text{top}, \text{ul}) \rangle. \tag{2.6}$$

In this case, we say A has a task-oriented intention to do s^* .

Thus, in our formalism here, the notion of a plan can, in principle, take a particularly degenerate form: it can be simply an assumption that some state s is the current state, plus an action sequence a^* that the agent has (somehow) determined (probably) lies in $N^*(A, s)$. This means that even though we understand intentions in terms of plans, we do not thereby require that agents have sophisticated planning capabilities in order to act contributively in their task-oriented activities.

The requirement that an agent must intend for its actions to make a coherent contribution to the task excludes, for example, randomly selected actions that happen to be coherent contributions to the task from counting as contributive. The (possible) presence of probabilistic reasoning in an agent's plan, however, means the agent's confidence in its plans can be graded. For example, an agent might perform a^* after estimating that there is a 30% or 60% or 95% chance that a^* is a coherent contribution in the current task state. We feel this aspect of our definition of a task-oriented intention is reasonable: agents can intentionally try to contribute to their tasks even if their plans for doing so are not foolproof.

2.5.5 Expectations of recognition

The second condition in our model requires that A have an expectation of recognition for its actions. The standard approach to achieving such an expectation is to require mutual belief about how the task is to be solved or completed, and further, about which agents are to take which actions at which points (Grosz and Sidner, 1990; Grosz and Kraus, 1996), and further, about all the various factors that might affect B 's interpretation process (Clark and Marshall, 1981). We relax these assumptions, and instead create a graded requirement that A have *some* sort of rationale that leads A to expect that B will recognize that A has taken the specific actions that A in fact took. This rationale *might* involve a model of mutual attitudes, but it also might follow from a less elaborate theory of B 's interpretation process, or even from a relatively shallow probabilistic model of A 's experience with being interpreted as intended.

To illustrate this approach, in our bookcase assembly task, let us consider how agent A might reason about whether B will recognize that A has performed the action sequence s^* in (2.6). First, suppose A knows that B will only observe A perform $\text{public}(A, s^*) = \langle \text{placeShelf}(\text{top}, \text{ul}) \rangle$. So B may or may not decide that A has also performed the tacit action $\text{insertBolts}(\text{ul})$. Let us consider two salient interpretations of $\text{public}(s^*)$ which B might attribute to A :

$$\begin{aligned} s^* &= \langle \text{insertBolts}(\text{ul}), \text{placeShelf}(\text{top}, \text{ul}) \rangle \\ f^* &= \langle \text{placeShelf}(\text{top}, \text{ul}) \rangle \end{aligned}$$

In the second interpretation, f^* , A has “forgotten” to insert the bolts, while in the first interpretation (the correct one), A remembered to insert them. How might A gain confidence that B will infer from observing $\text{placeShelf}(\text{top}, \text{ul})$ that A has not forgotten the bolts? There are any number of ways! Suppose A and B are outfitting their apartment with numerous identical bookcases, and this is the 6th bookcase they have assembled together. In assembling the previous 5 bookcases, A remembered to insert the bolts 4 times and forgot 1 time. In this case, A might assume B will assign:

$$\begin{aligned} P_B(I = s^* | O = \text{public}(A, s^*)) &= \frac{4}{5} \\ P_B(I = f^* | O = \text{public}(A, s^*)) &= \frac{1}{5} \end{aligned}$$

where I is a variable whose value is the complete action sequence A in fact performed, and the variable O represents B 's observation. If A makes this assumption, A has a substantive expectation of recognition for s^* , since A may reasonably expect that B will infer from observing $\text{public}(A, s^*)$ that A has (probably) done s^* (rather than f^*) as a contribution to completing their bookcase assembly task.

Though the expectation of recognition comes out probabilistic, we feel this intuitively captures what it means to try to make a clear contribution: when you act, you should have a reasonable expectation that you will be understood by your partner as doing what you in fact did, even if you are somewhat uncertain whether you will be. On the other hand, if you act without reasonably expecting your partner to recognize what action you took, you are not acting contributively in our sense.¹¹

This requirement is not without substance. For example, as a system builder, it is certainly possible to build an agent A whose run-time action choices do not take its partner B into account in

¹¹There may be good reasons for a generally contributive agent to not act contributively in specific circumstances. For example, there may be time pressure that prevents the formulation or execution of a contributive action. Or the agent's problem solving capacity may be insufficient to identify a contributive action, but sufficient to identify a non-contributive one. See Section 2.7 for further discussion of this issue.

any way. Suppose the agent is constructed to simply choose an action randomly from the alternative next actions it believes it could coherently perform in the task, and then to perform the action without any consideration of the public evidence A 's performance of the action will provide to B . At an intuitive level, it would be counterintuitive to describe such an agent as *expecting* for its action to be recognized by B . The more tacit actions that are present in the task, and the greater the degree of perceptual ambiguity that accompanies actions in the task, the less sense it would make to describe A as having this expectation.

Note that we leave open the possibility that the rationale that underlies an agent's expectation of recognition takes the form of an *intention* to be understood. While some agents may actually *plan* to be understood by their partners, our model does not require all contributive agents to do so. Similarly, our model requires only that A expect B to recognize the actions a^* that A has taken; it does not further require that A expect B to be able to recognize the task-oriented intention that motivated A to do a^* . This further assumption seems unmotivated, since A 's plan may depend on arbitrary aspects of A 's private experience and beliefs, including, for example, the exact probabilities A assigns to some state s being the current state, or to a^* being a coherent next contribution from that state, as in (2.5).¹² Thus we require that B be expected to recognize *what* A has done but not necessarily every last detail of why A did it.

2.5.6 Relation to deep coherence models

Two agents that act contributively in our sense will not necessarily have any *joint intentions*, a SharedPlan, or be engaged in a *shared cooperative activity*. This is because agents that satisfy our definitions may not have any mutual attitudes (beliefs, suppositions, or knowledge), which all of these models require (in one form or another). Similarly, the agents may not be engaged in a *joint activity* in the sense of Clark (1996, ch. 2), since they may not have achieved common ground about important aspects of their task.

Conversely, two agents that have a SharedPlan or that are engaged in a *shared cooperative activity* or *joint activity* will almost certainly satisfy our definition of contributive action when they act. The broad safety net of mutual attitudes, together with the agreement about the detailed plan for completing the task that these models require, make it a relatively simple matter for an agent to adopt a task-oriented intention with an expectation of recognition. It would take additional assumptions to characterize agents who have a *joint intention* as acting contributively in our sense,

¹²See also Stone (2004) for a detailed perspective on the relation between an agent's uncertainty and its communicative intentions.

because the *joint intentions* model mainly describes specific attitudes toward the team’s goals rather than how agents recognize each other’s individual contributions to those goals.

2.6 Contributive language use

In this section we refine the definition of contributive action developed in Section 2.5 to support contributions by way of natural language use in two-agent, task-oriented dialogue. Our general strategy remains the same: we develop definitions that characterize a use of language u by agent A as contributive if, and only if, in uttering u , A is trying to make a recognizable contribution to some two-agent task A is engaged in. The extension depends on two modeling assumptions that allow us to connect language to our model of two-agent tasks. We develop these modeling assumptions in the next two subsections.

2.6.1 Utterances generate dialogue acts

The first modeling assumption is that an utterance is fundamentally a type of *action*; see Clark (1996, p. 56-58) for discussion of the “language as action” and “language as product” traditions of language modeling. In particular, we assume that each utterance by a speaker *generates* (Goldman, 1970; Pollack, 1986) one or more *dialogue acts* (Bunt, 1994). In other words, by performing the action of uttering certain sounds (with an intention of a certain form, to be characterized shortly), a speaker thereby performs one or more dialogue acts.

We follow Harry Bunt (1994; 1996; 2000) in understanding a dialogue act as a “functional unit used by a speaker to change the context”. In particular, each dialogue act comprises a *semantic content* and a *communicative function*. The semantic content is the information the speaker is introducing into the context; for example, some proposition p . The communicative function is the way in which that information has been inserted in the context in order to play its intended role; for example, with a role like INFORM, YN-QUESTION, or CORRECT (Bunt, 2000). Together, the communicative function and semantic content determine an update function that maps the previous context to the new context that results from the dialogue act (Larsson and Traum, 2000). In computational terms, this conception of dialogue acts maps cleanly onto an inventory of typed dialogue actions parameterized by semantic contents.

We depart from Bunt (1996, 2000) and Morante et al. (2007), however, in that we do not define the context in terms of the mental states of the two interlocutors. Chapter 3 presents our alternative conception of the current context in a dialogue.

2.6.2 Dialogues are tasks

Our second modeling assumption is that the coherent patterns of dialogue acts that can occur in dialogue can be organized into *tasks* in the sense of Section 2.2. The notion of a “coherent pattern of dialogue acts” relates to various lines of research in dialogue modeling, including the study of *adjacency pairs*, the use of *dialogue grammars* to distinguish coherent sequences of dialogue acts (or speech acts) from incoherent ones, the use of state machines with dialogue acts (or speech acts) attached to the state transitions, and with plan-based models of dialogue coherence; see (Cohen, 1997) for a review of work using these techniques. From our perspective, our model of a task requires only that there be some function $N(A, s)$ that captures the set of alternative actions A can coherently perform next in state s . We take no stance on the structure of this function; thus our task model is consistent with all of these techniques, and others besides.

Some researchers have created taxonomies or inventories of the various kinds of dialogue acts that seem to occur in dialogue. For example, Bunt (1994) presents a hierarchical classification of dialogue acts that includes approximately 30 distinctions. There is a top-level distinction between *dialogue control acts* and *task-oriented acts* (on a narrower sense of *task-oriented* than ours here). Within the class of dialogue control acts, there are three subcategories: *feedback acts*, *discourse structuring acts*, and *interaction management acts*. And so on. Somewhat similarly, Traum and Hinkelman (1992) distinguish four top-level types of *conversation acts*: *turn-taking*, *grounding*, *core speech acts*, and *argumentation* (with the last being a multi-utterance concept). The DAMSL annotation scheme (Core and Allen, 1997) provides another hierarchical set of distinctions. For example, it distinguishes between the *backward* and *forward communicative functions* that *communicative acts* may have, and introduces further subcategories for each of these.

We believe all these distinctions reflect real phenomena in dialogue, but we don’t intend to characterize contributive language use explicitly in terms of such distinctions. Rather, we believe language users can use the same reasoning to make contributions *no matter what kind of dialogue act they are performing*. In particular, we will not characterize contributions using some set of distinguished “coordination-level dialogue acts” that coordinate the use of all the other types of dialogue acts.¹³ Rather, we subsume *all* of the kinds of acts (that a single utterance can generate) under the single heading *dialogue acts*, and characterize contributions to dialogue in terms of some global dialogue task that determines which dialogue acts can coherently happen next at each point.

Finally, note that nothing prevents a task in our sense from including a hierarchical organization

¹³Compare Traum (1994), in which grounding acts mediate the evolution of an incremental common ground representation.

of conceptually distinct subtasks.¹⁴ When subtasks are present, the various subtasks will interact in some way to determine what can coherently happen next in the overall dialogue. All but the most trivial dialogues are probably best analyzed in terms of such internal subtasks. However, we shall assume, without loss of generality, that there is a single global *dialogue task* in each two-agent, task-oriented dialogue. We then identify the *context* with the state of this global dialogue task. We thus use the terms *context* and *dialogue state* synonymously.

Figure 2.5 illustrates a simple example of a global dialogue task for bookcase assembly. This global dialogue task supports the original (completely non-verbal) route to success indicated in Figure 2.1 on page 12. But now we have also assumed that after any tacit action, the fact that the action has occurred can coherently be asserted through a dialogue act of the form `assert(past(Agent, Action))`. In this way, for example, an agent that performs a tacit action can tell its partner that it has done so. The effect of this assertion is to update the global dialogue state to include a record of the asserted proposition `past(Agent, Action)`. In this figure, we have indicated two such assertions that might occur after the tacit actions `insertDowels(u1)` and `insertBolts(u1)`.¹⁵ Space prevents us from sketching more fully the possible routes to a successful bookcase assembly following these assertions. The suggested approach, however, is for the global dialogue state to capture not only the state of the partially assembled bookcase, but also a record of all the propositions that have been asserted so far. In the next few sections, we will use this simple dialogue task to illustrate our model of how language use can be contributive in task-oriented dialogue.

2.6.3 Action sequences and dialogue acts

We now develop just a little notation for action sequences and dialogue acts before stating our model of contributive language use. We introduce a predicate `dialogueact(a)` which is true if and only if a is a dialogue act. We then define the possible sequences of dialogue acts $N_d^*(A, s)$ and of other acts $N_o^*(A, s)$ that A could coherently perform in dialogue state s :

$$\begin{aligned} N_d^*(A, s) &= \{d^* \mid d^* \in N^*(A, s) \wedge \forall d \in d^* \text{ dialogueact}(d)\} \\ N_o^*(A, s) &= \{o^* \mid o^* \in N^*(A, s) \wedge \forall o \in o^* \neg \text{dialogueact}(o)\} \cup \{\langle \rangle\} \end{aligned}$$

¹⁴The subtasks might organize the dialogue according to distinctions between different kinds of dialogue acts, such as those above. For example, as we will see in Chapter 3, COREF subsumes grounding related dialogue acts under an ambiguity management subtask that occurs frequently in its dialogues.

¹⁵In a more complete model, many more assertions and dialogue acts would be possible. This is an extremely simplified model of dialogue during bookcase assembly that we use to illustrate our approach without unnecessary complexity.

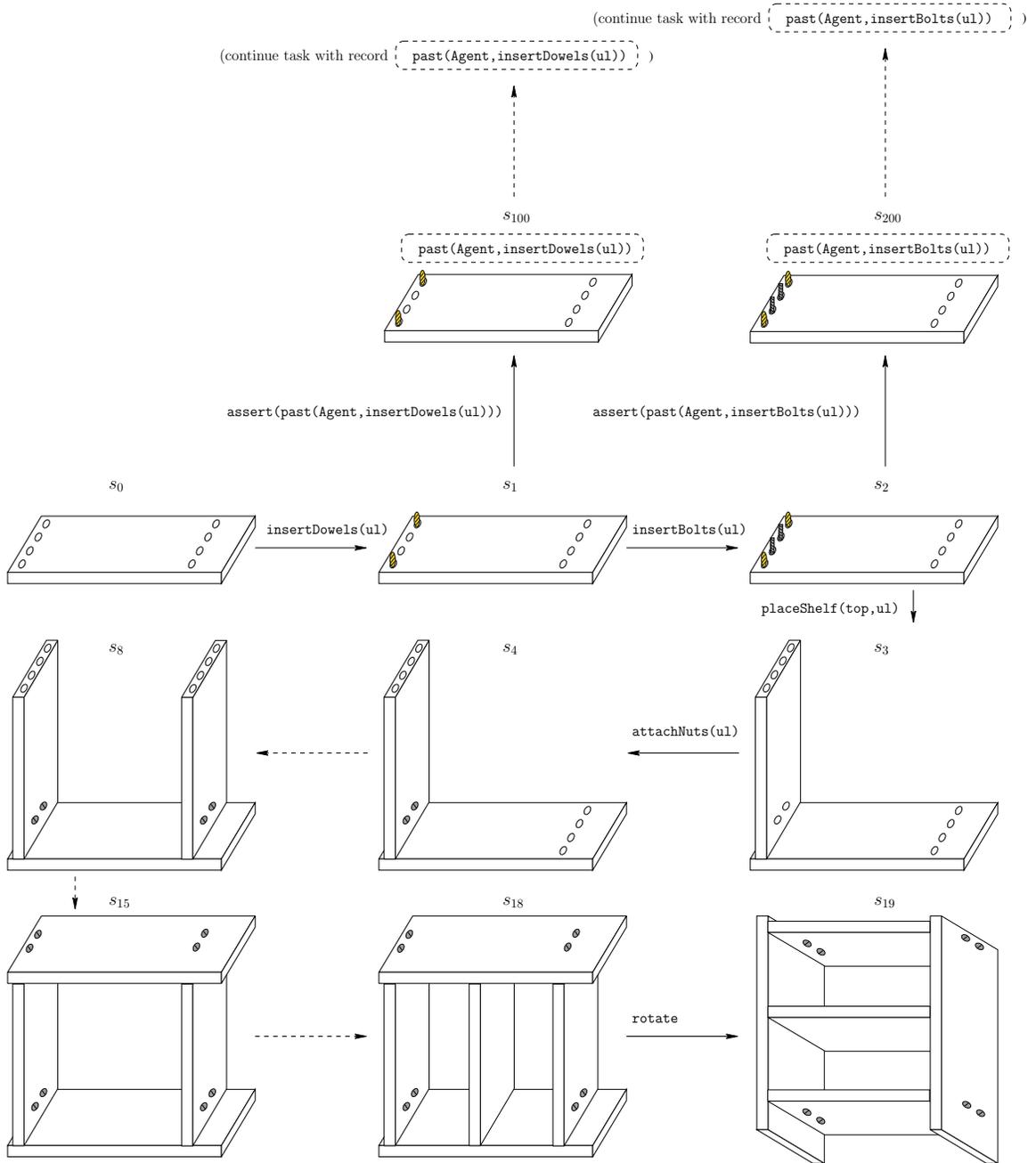


Figure 2.5: A simple model of bookcase assembly as a global dialogue task.

As in Section 2.5, we will continue to characterize contributions in terms of action *sequences*. We therefore generally aggregate an agent’s dialogue acts, in making an utterance, into a possibly longer sequence of actions. (In Chapter 3, the consideration of coherent action sequences will turn out to be an important aspect of COREF’s pragmatic reasoning about tacit actions.)

For simplicity, we assume that the dialogue acts an agent performs always occur as the *final* actions in such a sequence. To discuss these sequences, we will find the following notation convenient:

- (o^*, d^*) : a sequence of actions in which an agent does (zero or more) non dialogue acts o^* and then (somehow) performs dialogue acts d^* . For example, A might consider $(o^*, d^*) = (\langle \text{insertBolts}(\text{ul}) \rangle, \langle \text{assert}(\text{past}(A, \text{insertBolts}(\text{ul}))) \rangle)$.
- (o^*, u) : a “mixed” action sequence in which an agent does zero or more non dialogue actions o^* and then utters u to its partner. Such a sequence is “mixed” from the perspective of the dialogue task, since the actions o^* are part of the dialogue task while the exact utterance u is (presumably) not. For example, A might consider $(o^*, u) = (\langle \text{insertBolts}(\text{ul}) \rangle, \text{“I inserted the bolts”})$.
- (o^*, u, d^*) : a representation employed by an agent considering the possibility of doing o^* and then uttering u to its task partner to perform dialogue acts d^* . For example, A might consider $(o^*, u, d^*) = (\langle \text{insertBolts}(\text{ul}) \rangle, \text{“I inserted the bolts”}, \langle \text{assert}(\text{past}(A, \text{insertBolts}(\text{ul}))) \rangle)$.

2.6.4 Definition of contributive language use

With this notation in place, we now define contributive language use by A in a task-oriented activity with B .

- **Definition:** Agent A has a *task-oriented intention* to do (o^*, d^*) iff A intends to do (o^*, d^*) as part of a plan to advance the dialogue task, i.e.:
 1. in the (probabilistic) belief that the current dialogue state is some state s ,
 2. in the (probabilistic) belief that $o^* \in N_o^*(A, s)$,
 3. in the (probabilistic) belief that $d^* \in N_d^*(A, U^*(A, s, o^*))$.
- **Definition:** Agent A has an *expectation of recognition* for (o^*, u, d^*) iff A has a rationale according to which A expects (probabilistically) that B will infer from observing $\text{public}(A, o^*)$ and A ’s utterance of u that
 1. A has (probably) done actions (o^*, d^*) , and

2. A has (probably) done actions (o^*, d^*) in an attempt to advance the dialogue task.

• **Definition:** Agent A does (o^*, u) contributively iff there is some d^* such that

1. A has a task-oriented intention to do (o^*, d^*) ,
2. A has an expectation of recognition for (o^*, u, d^*) , and
3. A does (o^*, u) .

If an agent does some (o^*, u) contributively, we will describe A 's utterance of u as a contributive use of language.

Our model of contributive language use is entirely parallel to and consistent with our model of contributive non-verbal actions in Section 2.5. Where non-verbal action requires an intention to act with an expectation that your actions will be recognized, language use requires an intention to perform *dialogue* acts with an expectation that your dialogue acts will be recognized. The *only* difference in these models is that language use involves, by necessity, a translation between utterances and the dialogue acts they generate.

The model requires that a contributive use of language be motivated by a *task-oriented intention* to perform one or more dialogue acts d^* (optionally, after one or more non dialogue actions). As in the case of non-verbal action, we understand this intention in terms of a plan to make a coherent contribution to the two-agent task; the difference is that now the task is a *dialogue* task that interleaves dialogue acts with other kinds of actions. As part of this plan, the agent intends to perform one or more dialogue acts d^* . While actually performing these dialogue acts will of course require that some suitable words be uttered, we do not view the decision as to which words will be used to perform d^* as a prerequisite for the presence of the task-oriented intention to perform d^* . The presence of a *task-oriented intention* simply means the agent has a plan to perform some dialogue acts, one way or another.

For example, in the bookcase assembly dialogue task of Figure 2.5 on page 44, suppose A 's mental state includes the following assessments:

$$\begin{aligned}
 P(\text{state} = s_1) &= 0.9942 \\
 P(\langle \text{insertBolts}(\text{ul}) \rangle \in N_o^*(A, s_1)) &= 0.8650 \\
 P(\langle \text{assert}(\text{past}(A, \text{insertBolts}(\text{ul}))) \rangle \\
 \in N_d^*(A, U^*(A, s_1, \langle \text{insertBolts}(\text{ul}) \rangle))) &= 0.8281.
 \end{aligned} \tag{2.7}$$

Partly on the basis of these assessments, suppose A adopts the intention to carry out the action sequence:

$$(o_1^*, d_1^*) = (\langle \text{insertBolts}(\text{ul}) \rangle, \langle \text{assert}(\text{past}(A, \text{insertBolts}(\text{ul}))) \rangle). \quad (2.8)$$

In this case, we say A has a task-oriented intention to do (o_1^*, d_1^*) .

For a use of language to be contributive, in addition to having a task-oriented intention, the speaking agent must also have an expectation that the dialogue acts it intends to perform will (probably) be recognized by its partner. This again mirrors our model of non-verbal action: to speak contributively, you must have a reasonable expectation that you will be understood by your partner as performing the dialogue acts you in fact did perform, even if you are somewhat uncertain whether you will be. On the other hand, if you speak without expecting your partner to recognize what dialogue acts you are performing, you are not speaking contributively.

As with non-verbal action, the requirement that a contributive language user must have an expectation of recognition for its dialogue acts is not without substance. For example, an agent that consults a static look-up table of “canned text” (Reiter and Dale, 2000, p. 68) in order to express its intended dialogue acts in words is difficult to view as rationally “expecting” its dialogue acts to be recognized. If the canned text has been carefully chosen for the task, perhaps the agent’s intended dialogue acts may somehow turn out to always be recognized correctly by its human addressees, but, intuitively, this would be more an engineering success story than a reflection of the agent’s ability to coordinate to make sure its contributions are recognized.

As an example of how a practical agent might assess the probability of recognition for an utterance, let’s return again our bookcase assembly task. Suppose agent A has adopted the intention (o_1^*, d_1^*) indicated in (2.8) on the current page. Since the `insertBolts(ul)` action is tacit, A knows that B will observe only the utterance that A chooses to utter. Let us suppose A is considering uttering “I inserted the little parts” as a way of generating the dialogue act `assert(past(A, insertBolts(ul)))`. According to our model of contributive language use, A should consider whether this utterance will lead B to infer that A has done (o_1^*, d_1^*) rather than some other action sequence. In this instance, A might notice that due to an ambiguity in “the little parts”, which might mean the dowels or the bolts, there is a possible distractor interpretation in which “I inserted the little parts” is interpreted as generating `assert(past(A, insertDowels(ul)))` rather than `assert(past(A, insertBolts(ul)))`. What is worse, from the current state s_1 , the dialogue task of Figure 2.5 supports both these interpretations, by way of the following action sequences:

$$\begin{aligned}
(o_1^*, d_1^*) &= (\langle \langle \text{insertBolts}(\text{ul}) \rangle, \langle \text{assert}(\text{past}(A, \text{insertBolts}(\text{ul}))) \rangle \rangle) \\
(o_2^*, d_2^*) &= (\langle \langle \text{assert}(\text{past}(A, \text{insertDowels}(\text{ul}))) \rangle \rangle)
\end{aligned}$$

I.e., in uttering “I inserted the little parts”, A might be asserting the previous addition of the wooden dowels, or A might have inserted the bolts and be reporting *that* insertion. Perhaps A ’s lifetime of dialogue experience suggests B would assess these interpretations as follows:

$$\begin{aligned}
P_B(I = (o_1^*, d_1^*) | O = \text{“I inserted the little parts”}) &= \frac{1}{2} \\
P_B(I = (o_2^*, d_2^*) | O = \text{“I inserted the little parts”}) &= \frac{1}{2}
\end{aligned}$$

In this case, A does not have a very strong expectation of recognition for the task-oriented intention (o_1^*, d_1^*) , because another interpretation is just as likely. In order to speak more recognizably, A might choose instead to utter “I inserted the bolts”, with this superior expectation of recognition:

$$\begin{aligned}
P_B(I = (o_1^*, d_1^*) | O = \text{“I inserted the bolts”}) &= 0.94 \\
P_B(I = (o_2^*, d_2^*) | O = \text{“I inserted the bolts”}) &= 0.06
\end{aligned}$$

This example illustrates an important feature of our model of contributive language use, which is that expectations of recognition must be assessed with respect to tacit actions as well as public actions. In this way, an agent’s coordination reasoning encompasses unobservable events generally, including tacit mental events which sometimes need to be communicated in dialogue tasks. We discuss COREF’s general strategy for speaking contributively in the presence of tacit actions in Chapter 3.

Note that, as with non-verbal action, we have again left open the possibility that the rationale that underlies an agent’s expectation of recognition takes the form of an *intention* to be understood. One way an agent could intend to be understood would be to plan out its exact utterance so that its intended dialogue acts will be uniquely recognizable in the specific context (dialogue state) the agent believes it is in, as in (Stone et al., 2003; Stone, 2004; DeVault et al., 2004). We will see in Chapter 3 that COREF plans its utterances using reasoning along these lines.

Of course, an utterance that is intended to be uniquely recognizable in a specific context is not necessarily uniquely recognizable by the addressee of the utterance: the addressee may not think that is the context they are in, or may make different assumptions in interpretation. This is the point at which assumptions of common ground or mutual attitudes work their magic: if the two interlocutors have attained (hierarchical) agreement about the context, and about all the factors

that affect interpretation in context, then the unique interpretation that the speaker has in mind will surely be the interpretation the hearer assigns to the utterance.

By contrast, our model of contributive language use acknowledges and accommodates the risk of disagreement about the context. A speaker need not assume common ground to formulate a *probabilistic* expectation of recognition. Rather, the speaker needs *some* kind of rationale that bounds or hedges the probability of misunderstanding due to a disagreement about the current context or interpretation. Chapter 3 sets out a model of uncertainty about context that allows agents to formulate robust utterances under this kind of uncertainty, and shows how COREF does so as an illustration of the theory.

There are various approaches to natural language generation that amount to intermediate kinds of expectations of recognition. For example, an agent using template-based generation (Reiter, 1995) could use aspects of context to choose its run-time utterances. Depending on the details, such an agent might defensibly be described as trying to formulate a recognizable contribution for its addressee. The case is somewhat unclear however, since in a system that uses template-based generation, there is almost certainly no consideration of alternative utterances that might be *more* recognizable expressions of the same dialogue act(s).

Systems that use statistical observations to choose utterances that lead to greater “reward signals” in training interactions with users are another intermediate case (see e.g. Williams and Young, 2007). The richer the variety of utterances such a system considers, the more persuasively one might argue that the system is selecting a context-specific utterance that it thinks will be understood as the system “intends”; however, the question of what such a system’s “intention” is, if anything, and the connection between being understood and the reward signal may not be sufficiently clear for such a case to be articulated.

2.6.5 Other language use

We do not claim that all human language use is contributive in the sense articulated here. As discussed further in Section 2.7, there may be real-world situations in which a speaker is unable to speak contributively when they speak. A human speaker may also sometimes deliberately speak in a way that is not contributive. For example, in an “backhanded compliment”, *A* may intend to insult *B*, but select words that *A* expects to leave *B* uncertain whether *A* has insulted or complimented *B*. If the distinction between an insult and a compliment shows up as a distinction in the dialogue act performed, or the content expressed, then this is non-contributive in our sense.

More generally, speaking without intending to make a recognizable contribution may sometimes serve useful purposes in human-human dialogue. For example, certain kinds of humor, negotiation strategies, and conversational genres such as flirting, in which linguistic choices may be socially consequential, risky, or taboo, seem to involve the selection of utterances that are deliberately ambiguous.

Finally, there may be occasions when a speaker's uncertainty about the current context leads them to select an utterance that is essentially contributive, but is such that the precise dialogue acts to be performed and recognized are underspecified. We discuss an example of this kind in Chapter 3.

2.7 Interleaving contributive actions with other actions

The models of contributive action in Sections 2.5 and 2.6 suggest a dynamic in which a task is completed through a series of successive contributive actions by the two participants. But it may sometimes happen, for one reason or another, that the agents act individually to advance their task in ways that are not contributive in our sense. For example, there may be time pressure that prevents the formulation or execution of a contributive action. Or the agent's problem solving capacity may be insufficient to identify a contributive action, but sufficient to identify a non-contributive one. Or the nature of the task may be such that *A* must take a series of actions tacitly, as when *B* cannot see *A* for an extended period.

We give an example of the latter scenario in Thomason et al. (2006, p. 15):

Andrew and Bess are cooking. Bess asks Andrew to wash and dry some spinach for a salad. Bess then turns her back to Andrew and attends to a pot on the stove. It is boiling ferociously, and she can neither see nor hear what Andrew does. Andrew cleans the spinach. Bess turns to face Andrew again.

...

In plain view, Andrew mixes the spinach with the other salad ingredients. In fact, this action furthers the joint activity in a straightforward way; Andrew and Bess are one step closer to a good salad.

In that paper, we go on to characterize the inference that allows Bess to infer that Andrew has tacitly washed the salad in terms of what we call an ENLIGHTENED UPDATE to their common ground. The model of contributive action we have presented here would narrate this inference similarly.¹⁶

¹⁶Note however that our new definitions do not make the fact that Andrew's public mixing of the spinach with the other salad ingredients is contributive dependent on the (eventual) (or intended) presence of common ground about whether Andrew has washed the spinach. Perhaps Andrew has been known on occasion to ignore Bess's requests to wash things; the issue has been a point of heated contention between the two. Though Andrew has promised not to deceive Bess again, both are aware that Bess retains a lingering suspicion about Andrew's commitment in this matter. The present model of contributive action does not force the issue: Andrew's tacit washing and public mixing of the spinach can be seen as contributive so long as Andrew has a reasonable expectation that Bess will infer from observing the mixing that Andrew has done both.

The example does however highlight an important latent assumption in the new model of contributive action and language use in Sections 2.5 and 2.6. If we consider the aggregate sequence a^* of *all* the task actions an agent performs as part of an extended two-agent task-oriented activity, there is a question as to how the agent’s internal planning process partitions up the aggregate sequence into a series of individual task-oriented intentions. That is, there is presumably some partition $a^* = (a_1^*, \dots, a_n^*)$ such that the agent formulates each task-oriented intention a_i^* separately.

For example, there is a question as to whether, at the moment Andrew formulated his (task-oriented) intention I to (tacitly) wash the spinach, that same intention I also included his subsequent public mixing of the spinach before Bess. If so, and if in formulating I , Andrew further expected Bess to infer from the planned public mixing that Andrew has washed the spinach, then Andrew straightforwardly satisfies our definition of contributive action in Section 2.5.

However, there is no reason to assume, in general, that an agent’s individual planning episodes will always yield an intention that motivates a contributive action in the sense of Section 2.5. For example, Andrew might simply formulate the task-oriented intention W to perform the single action of washing the spinach, without considering at that time how Bess will eventually learn that he has done so. According to our current definitions, Andrew does not act contributively in executing W . Though he *is* acting to advance his two-agent task with Bess, since she will not observe the tacit washing, Andrew cannot (yet) expect her to recognize that it has occurred.

When Andrew subsequently mixes the spinach, our current definition of contributive action does not perspicuously require that Andrew expect that Bess will infer that he has washed it as well. To make this expectation explicit, we could extend our definition as follows:

- **Definition:** Agent A does a^* contributively¹ iff
 1. A has a task-oriented intention to do a^* ,
 2. A has an expectation of recognition for (t^*, a^*) ,
 where t^* is the sequence of (zero or more) tacit actions A has performed prior to a^* but subsequent to any previous public action by A , and
 3. A does a^*

Parallel reasoning would lead us to extend our definition of contributive language use as well:

- **Definition:** Agent A does (o^*, u) contributively¹ iff there is some d^* such that
 1. A has a task-oriented intention to do (o^*, d^*) ,

2. A has an expectation of recognition for $((t^*, o^*), u, d^*)$,
 where t^* is the sequence of (zero or more) tacit actions A has performed prior to (o^*, u)
 but subsequent to any previous public action by A , and
3. A does (o^*, u) .

The effect of these amendments is to require that each new *contributive* action or utterance by A be expected to bring the other agent B “up to sync” about any relevant tacit actions A has performed since A ’s last public action.

More generally, we could acknowledge that A may have performed some *public* actions in a non-contributive way,¹⁷ and require that A bring B up to sync on those as well:

- **Definition:** Agent A does a^* contributively² iff
 1. A has a task-oriented intention to do a^* ,
 2. A has an expectation of recognition for (n^*, a^*) ,
 where n^* is the sequence of (zero or more) non-contributive actions A has performed prior
 to a^* but subsequent to any previous contributive action by A , and
 3. A does a^*

And similarly:

- **Definition:** Agent A does (o^*, u) contributively² iff there is some d^* such that
 1. A has a task-oriented intention to do (o^*, d^*) ,
 2. A has an expectation of recognition for $((n^*, o^*), u, d^*)$,
 where n^* is the sequence of (zero or more) non-contributive actions A has performed prior
 to (o^*, u) but subsequent to any previous contributive action by A , and
 3. A does (o^*, u) .

We see it as a design question for a system builder which of these (now three) parallel notions of contributive action is implemented. For example, as we will see in later chapters, COREF implements the original definitions of Sections 2.5 and 2.6. It is able to do so because it is possible, in COREF’s task, for each public action by each player to be associated with a single contribution that

¹⁷For example, to poke fun at the simmering tension regarding Andrew’s commitment to washing things properly, Andrew might have made a special effort to let Bess see him at the sink making exaggerated “spinach-washing motions”, but taken care not to let her see any actual spinach being washed. I.e., Andrew may have intentionally left the occurrence of the spinach washing action ambiguous in an effort to provoke Bess. On our model, Andrew’s mischievousness here makes his spinach washing non-contributive (even if he does it).

incorporates all the tacit actions that may have preceded it. Agents that coordinate in other tasks and domains, salad-making perhaps, may require a more flexible relationship between intentions and contributive action.

2.8 Relation to interlocutor mental states

We have given definitions that allow an uncertain agent to try to make coherent, recognizable contributions to their two-agent tasks. In comparison to deep coherence models such as SharedPlans, *joint intentions*, and *shared cooperative activity* (see Section 2.3.2), this conception suggests that acting contributively is relatively independent of detailed reasoning about how your addressee’s mental state may change, or has changed, as a result of your utterances.

Of course, in general, each dialogue act performed by agent *A*, in an effort to advance the dialogue task, will have some effects on agent *B*’s mental state. In some cases, presumably including many human-human utterances, these effects may be explicitly *intended*. This raises the question of how our framework might comport with a view of utterance planning that explicitly connects the utterance not just to its update to the two-agent dialogue state but also to planned effects on the hearer’s mental state. In particular, how might the communicative intention of a contributive agent incorporate these intended effects?

In the next two sections, we survey a couple of ways our model of contributive action might be connected to existing pragmatic theories that link speaker intentions to planned changes to the hearer’s mental state.

2.8.1 Relation to Gricean intentions

One particularly influential theoretical connection between a speaker’s communicative intention and planned changes to the hearer’s mental state is Grice’s view of *nonnatural meaning* (Grice, 1957). According to Grice, a speaker *A* has a specific meaning in making an utterance iff the speaker’s utterance is motivated by a particular kind of complex intention regarding how the addressee will reason about the utterance. For example, one situation in which a speaker *A* can be said to nonnaturally mean some proposition *p* in uttering something to *B* is when the following three conditions are satisfied:

1. *A* intends for *B* to believe *p*,
2. *A* intends for *B* to recognize *A*’s intention in 1, and

3. A intends for B 's recognition in 2 to be part of B 's reason for believing p .

These conditions are designed to single out Grice's concept of nonnatural meaning, which he argues is what distinguishes a speaker's meaning in a use of language from other kinds of *natural meaning* that people can attribute to observations, actions, or events.

In comparison to Grice's theory, our model of contributive language use makes weaker assumptions about the speaker's intentions regarding his addressee. Our model requires only that the speaker expect that his task-oriented intention will be recognized by the addressee; it does not further require that the speaker intend for this recognition to result in an update to the hearer's mental state as detailed in Grice's theory.

However, our model of contributive language use is consistent with Grice's theory. We can relate our model to nonnatural meaning as follows:

- A means_{NN} that p by a contributive use of u with task-oriented intention (o^*, d^*) iff

1. A intends for B to believe p .

◊ For example, A might intend B to assign $P(p) > \frac{1}{2}$.

2. A intends for B to recognize A 's intention in 1.

◊ For example, A might have a plan for B to employ the following line of reasoning:

(a) A intended (o^*, d^*) by u .

(b) $P(p|A \text{ intended } (o^*, d^*)) > \frac{1}{2}$, so

(c) A intends me to assign $P(p) > \frac{1}{2}$.

3. A intends for B 's recognition in 2 to be part of B 's reason for believing p .

◊ For example, A might intend B to employ the following line of reasoning:

(a) A intends me to assign $P(p) > \frac{1}{2}$, so

(b) $P(p) > \frac{1}{2}$.

In other words, our model makes contributive language use independent of, but consistent with, Gricean non-natural meaning. In particular, it is possible to build an agent that speaks contributively, in our sense, but does not mean_{NN} anything by its utterances. Indeed, as we will see in Chapter 3, our implemented agent COREF is (arguably) such an agent. We view this possibility as a strength of our practical model; while it remains consistent with the presence of nonnatural meaning in dialogue, it does not require that builders of contributive agents implement the more detailed planning and reasoning about hearer mental state that meaning_{NN} requires.

2.8.2 Relation to speech act theory

Speech act theory (Austin, 1962; Searle, 1969) is another prominent approach to connecting a speaker's communicative intentions to planned changes to the addressee's mental state. This approach views each utterance by a speaker as an action, fundamentally, rather than merely an expression of a proposition or a transmission of information to the hearer. The theory suggests that there is a range of *speech acts* which speakers intend to perform in speaking, and which hearers try to recognize in interpretation.

However, there is still no generally agreed inventory of clearly delineated speech acts that can be straightforwardly implemented in dialogue systems (Traum, 1999a).¹⁸ One problem is that the range of possible speech acts has proved difficult to circumscribe theoretically. Instead, researchers typically rely on intuitive examples of speech acts like *ask*, *assert*, *warn*, *request*, *inform*, *promise*, *threaten*, and many others. The general tenor of these examples is that speech acts correspond to English verbs that describe things speakers can do (or try to do) by making an utterance.

Another important issue is that the various speech acts that are commonly postulated seem to depend in different ways, and to different extents, on the cognitive impacts the utterance may have on addressee. Some candidate speech acts like *persuade* or *scare* seem inextricably linked to the changes to the addressee's mental state that *actually result* from the utterance. These changes are known as *perlocutionary effects* of the utterance, and speech acts which seem inextricably linked to these effects are known as *perlocutionary acts* (Austin, 1962).

Another class of speech acts, *illocutionary acts* (Austin, 1962), seem to vary in their dependence on the addressee's mental state. For example, intuitively, a speaker can *assert* or *state* some proposition *p* even if the addressee mistakes the assertion or statement for a question, or if the addressee interprets the speaker as asserting *q* rather than *p*, and possibly even if the addressee is distracted and fails to hear the utterance at all. On the other hand, again intuitively, a speaker cannot *inform* his addressee that *p* unless the addressee hears the utterance, identifies *p* correctly, and (perhaps) also correctly recognizes that the speaker is informing him that *p* rather than, say, asking or promising.

Such observations suggest a number of theoretical questions about the occurrence of speech acts in dialogue:¹⁹

1. Does the performance of a speech act require that the hearer correctly recognize the speaker's intended speech act?

¹⁸See Traum (1999a) for a thorough review of the history of speech acts in dialogue systems, and alternative perspectives on how utterances can be related to actions.

¹⁹See Traum (1999a) for further discussion.

2. Does the performance of a speech act require that the hearer correctly recognize additional aspects of the speaker's communicative intention?
3. Does the performance of a speech act require specific changes to hearer mental state?
4. Does the performance of a speech act require hearer confirmation that recognition of the speaker's intention was unproblematic?

Researchers have answered these questions in different ways, and have put forth correspondingly different theories of speech acts. Among the positions that have been advocated are these:

- The performance of a speech act depends only on the speaker. Speech acts occur independently of the hearer's interpretation of them, or reaction to them. See, e.g., Cohen and Levesque's (1990a) model of illocutionary acts as "attempts", or Perrault and Allen's (1980) definition of the performance of an illocutionary speech act.
- While the performance of a speech act depends only on the speaker, the *successful* performance of a speech act requires that the speaker intend for the hearer to recognize the speaker's intended speech act, and that the hearer correctly do so. See Cohen and Levesque (1990b) for discussion of this view.
- While the performance of a speech act depends only on the speaker, speech acts are attempts to change the hearer's mental state (including possibly hearer intentions to act and mutual attitudes); as such, the *successful* performance of a speech act depends on specific updates to the hearer's mental state rather than correct intention recognition. This view includes Cohen and Levesque (1990a,b) and (arguably) Perrault and Allen (1980).
- The performance of a speech act is inherently a multi-agent action. The hearer must confirm unproblematic recognition of the speaker's attempted speech act through a grounding action. See, e.g., Traum and Hinkelman (1992) and Traum (1994).

Theories of speech acts thus differ in their conceptions of speech acts as unilateral or multi-agent actions, and in the ways they frame a successful speech act in terms of hearer mental state and hearer followup actions.

We believe our model of contributive language use is consistent, more or less, with all these views of speech acts. On a unilateral view of speech acts, speech acts can simply be identified with (some of) the dialogue acts a speaker performs in making an utterance. The ultimate success or failure of

the speech act, then, could be defined in terms of the hearer’s correct recognition of the speaker’s speech act, or in terms of the relationship between the hearer’s subsequent mental state and the speaker’s speech act.

On a multi-agent view of speech acts, speakers would need to be viewed not as directly performing speech acts but rather as incrementally performing coordination-oriented dialogue acts such as presenting a speech act, acknowledging a speech act, etc., as in Traum (1994). Here, success can be understood either in terms of the coordination acts that occur during this multi-utterance grounding process, or in terms of the interlocutors’ mental states as the grounding process evolves.

Thus, our model of contributive language use does not force any particular theoretical view of dialogue in terms of speech acts. Our own current view is closest to a unilateral view of a speaker’s dialogue acts, with success (implicitly) understood as the (eventual) correct recognition by the hearer of the speaker’s original intention. However, we are not committed to any particular theoretical analysis of dialogues in terms of speech acts.

2.9 Conclusion

In this chapter, we have presented contributive action as a practical target for uncertain collaborative agents. Acting and speaking contributively involves choosing actions that contribute to a task with an expectation that those actions will be recognized by your task partner. While we have framed the reasoning by which an agent can aim for understanding in explicitly probabilistic terms, we have yet to spell out in detail how an agent can manage its uncertainty as a dialogue unfolds over time. In the next chapter, we expand our perspective to include an uncertain agent’s reasoning across multiple turns in dialogue. We use contributive action as a conceptual core in a reasoning process we call *contribution tracking*, which allows interlocutors to act contributively despite uncertainty that may persist across several dialogue turns. Contribution tracking connects our work with a range of work in pragmatics which sees the back-and-forth of language use in dialogue as a collaboration, and which talks more precisely about the processes of interpretation and generation that allow us to understand one another. In the next chapter, we show how to model these processes in a particular domain using specific task models, specific models of grammar, specific algorithms for choosing contributions, and specific algorithms for tracking contributions over time. This shows how we can exactly implement the theoretical model presented in this chapter and helps to substantiate our arguments in favor of contributive action as well as contribution tracking.

2.9.1 Limitations, caveats, objections

In this section we discuss a few possible objections to the conception of contributive action and language use we have developed in this chapter.

Not all dialogue is task-oriented! What about general conversation?

While we have not attempted to implement a computational model of “general conversation”, our hunch is that general conversation is in fact mostly task-oriented in the sense of Section 2.2. The intuition here is that general conversation is not just “saying anything at any point until you get bored”, but rather being flexible about which tasks are undertaken in subdialogues, and in what order. In particular, the empirical question would be whether all the utterances within a general conversation dialogue can be covered by task-oriented analyses of subdialogues. But to make this case, we would need to go beyond the kinds of task models that are currently implemented in agents like COREF to include a wider range of more social tasks (e.g., *A* finds out what *B* knows about *X*, *A* and *B* gossip freely about *Y*, *A* conveys to *B* the recent events in *A*’s life that *A* thinks are important, etc.). Such an inventory of social tasks would help us better evaluate the extent to which there are constraints on what dialogue acts can coherently happen next in general conversation, and thus whether our model of contributive action in task-oriented dialogue applies. But in any case, it is not obvious that general conversation is not *task-oriented* under our permissive formalization of that term.

Not all dialogue is collaborative! What about “adversarial” or “competitive” dialogue?

There are two main ways a two-agent dialogue can involve language use that isn’t contributive in our sense.

1. There can be no dialogue task that the agents are trying to coherently advance with their utterances.
2. The agents can expect their intended contributions to the task not to be recognized.

One way dialogue can be adversarial or competitive would be for the interlocutors to push distinct dialogue agendas (subtasks), and refuse to contribute to subtasks introduced by their dialogue partners:

A: where *WERE* you last night?
 B: where were *YOU* this morning?
 A: tell me where you *WERE* last night!
 B: tell me where *YOU* were this morning!
 ...

In examples like this, “adversarial” interlocutors may engage in a turn-by-turn tug of war over which dialogue subtask will be taken up next. However, each speaker seems to intend for their proposed subtask to be recognized unambiguously, and thus can be viewed as acting contributively, when they speak. Additionally, our approach to modeling tacit actions in dialogue, developed in Chapter 3, could in principle contribute to explaining how a speaker can try to be understood while tacitly shifting the task in ways that are adversarial or competitive. This idea is not pursued further in this dissertation, however.

Another way of being adversarial is to lie to your dialogue partner. This issue is again independent of contributive action in our sense. A speaker can lie while speaking contributively by intending to perform a dialogue act that contributes coherently to a current dialogue task, and expecting that a specific dialogue act will be recognized, as *B* does here:

A: where *WERE* you last night?
 B: at my parents’ house.
 [*B* was not at *B*’s parents’ house.]

B’s contributive answer here is a lie because *B* intends *A* to come to believe the contributed answer, which contradicts *B*’s beliefs.

Another way of being adversarial or competitive is to insult or try to intimidate your addressee. Both of these can be done with language that is contributive in our sense.

Finally, in adversarial or competitive dialogues, the interlocutors may have very different private goals as they pursue a common dialogue task together (Clark, 1996, p. 34-35). This may result in some of the above types of adversarial behavior in dialogue, but, again, this is an independent issue from contributive language use in our sense.

What about when the understanding of the task differs between agents?

Some of the complex conditions involving mutual attitudes in deep coherence models, especially in SharedPlans, are designed to prevent problematic discrepancies in agents’ private models of a task

from interfering with a successful task outcome. Certainly, if two agents have mutual belief or mutual knowledge about their task, including the different ways of solving it, the way they intend to solve it on a particular occasion, and the criteria of success, then they can avoid many potential coordination failures. However, we take a different approach, and do not try to achieve this bedrock of successful coordination directly in our characterization of contributive action. Our general attitude toward these issues has two main dimensions.

First, our general view is that coming to agreement on the structure of a task, the different ways of solving it, how to solve it on a particular occasion, and the criteria of success, are *themselves* two-agent, task-oriented activities which agents must solve with contributive action and language use. So while we have not, to date, attempted to model complex domains in which agents' task models differ in ways that are problematic, our approach in such a case would be to try to reduce the overall interaction to one of contributive action within a task negotiation "meta task", followed by contributive action (in our sense) on the resulting first-order task. To avoid a regress, this response seems to commit us to there being a certain basic inventory of tasks which all dialogue agents can take for granted (as identically understood between agents). We believe this is not implausible; for example, common dialogue subtasks such as yes/no questions, wh-questions, etc., might plausibly be common to all language-using agents. However, developing such an inventory of basic tasks is well beyond the scope of this work.

Second, the model of uncertainty about dialogue state that we develop in Chapter 3 grants us some flexibility in narrating the reasoning processes of two dialogue agents who seem to disagree about the task they are engaged in. In particular, in our model, an agent may lose track of what the current dialogue task is, or it may face specific uncertainty about what the current dialogue task is.

Chapter 3

Contribution tracking in two-agent, task-oriented dialogue

This chapter presents *contribution tracking* as a reasoning process by which dialogue agents can track the contributions that are being made across multiple utterances and actions in task-oriented dialogue. Our presentation of contribution tracking builds on the characterization of isolated contributive actions in Chapter 2 by developing a model of an agent’s reasoning and uncertainty about dialogue state in a multi-utterance dialogue setting. We illustrate contribution tracking using COREF (DeVault et al., 2005; DeVault and Stone, 2006; DeVault et al., 2006; DeVault and Stone, 2007), an implemented dialogue agent that tracks contributions while participating in an object identification game under uncertainty.

We begin in Section 3.1 by defining what we mean by *contribution tracking*. In Section 3.2, we discuss the collaborative view of reference in dialogue that motivated COREF’s development as a research project, and present the object identification game that COREF plays. In Section 3.3, we present COREF’s internal design as a particular instantiation of a more general modular architecture which we call the RUBRIC system architecture. We present the RUBRIC architecture as a general modular approach to building dialogue systems that track contributions in a two-agent dialogue task. This architecture constitutes a way of circumscribing the reasoning and representations that an implemented agent needs to employ in order to track contributions, and of organizing this reasoning into a set of specific problem-solving modules. We show how this architecture enables COREF to speak and act contributively in its object identification game, and argue that this approach to system building offers several advantages over alternative designs for collaborative dialogue systems.

In this chapter, we will use a few examples to present and explain COREF’s detailed pragmatic reasoning in dialogue. We delay a more thorough empirical evaluation of COREF to Chapter 4.

3.1 Contribution tracking

In this section, we motivate and define contribution tracking as a reasoning process that can be carried out by uncertain dialogue agents.

3.1.1 Motivation

Contribution tracking is an attempt to make sense, in a probabilistic setting, of three fundamental claims made by mainstream theories of pragmatic reasoning in human-human dialogue:

1. interlocutors track and exploit the evolving context to coordinate their individual contributions;
2. the current context depends on what the previous utterances of both interlocutors have meant (contributed);
3. what a speaker can recognizably mean (contribute) by a specific choice of words depends on the current context.

The first claim is essentially the framework developed in Stalnaker (1974, 1978). Stalnaker’s work has provided a basic conceptual vocabulary for a wide range of subsequent work that attempts to construct knowledge level descriptions of linguistic interpretation in detailed dialogue contexts (e.g. Clark and Marshall, 1981; Thomason, 1990; Clark, 1996; Poesio and Traum, 1997; Thomason et al., 2006). Stalnaker describes communication as taking place “against a background of beliefs or assumptions which are shared by the speaker and his audience, and which are recognized by them to be so shared” (1974). Stalnaker describes these speaker presuppositions as what the speaker takes to be the “common ground” between the interlocutors. A speaker uses this common ground to achieve efficient communication — for example, there is no need to assert a proposition that is already common ground — and as a way to keep track of the current state of the dialogue — by treating a new assertion as causing a specific update to the common ground between the interlocutors, for example. Thus, in Stalnaker’s framework, interlocutors in dialogue track and exploit their evolving common ground (which we, here, would call the *context*) in order to coordinate their individual contributions.

The second claim is the basic framework of linguistically motivated dynamic semantic theories (e.g. Kamp, 1981; Heim, 1982; Kamp and Reyle, 1993; Poesio and Traum, 1997), which formalize the effects of utterances in dialogue as specific updates to the interlocutors’ shared dialogue context.

The third claim is embodied in a range of work on the pragmatic interpretation of utterances, including formal work in dynamic semantics, as well as computational approaches to diverse topics such as referring expression generation in context (e.g. Dale and Reiter, 1995), anaphora resolution in dialogue systems (e.g. Byron, 2002), and generation of complete utterances that convey a recognizable communicative intention in context (e.g. Stone et al., 2003).

While these three claims capture important insights about natural language use in context, it is not obvious how they can be translated into a probabilistic setting. For example, as we discuss above in Section 2.4.4 (page 30), a notion of context based on mutual attitudes, such as Stalnaker’s common ground, is not easy to treat probabilistically in an implemented system. This difficulty is also one impediment to the computational implementation of detailed dynamic semantic theories which view the evolving dialogue context in terms of mutual attitudes or common ground (e.g. Beaver, 2001; Barker, 2002). Nor is it easy to see how many existing computational models of interpretation in context should be adapted to a situation in which a speaker is uncertain about what the context is. For example, referring expression generation is typically framed as a problem of distinguishing an intended referent from the distractor objects in a specific context (Dale and Reiter, 1995; DeVault et al., 2004). But what if a speaker in dialogue is uncertain about the context in a way that affects her determination of what the distractor objects are? How shall she generate an appropriate referring expression for a specific object? Similarly, if an anaphora resolution algorithm predicts that an anaphoric expression would be resolved one way in one context, but another way in another context, how can an interlocutor who is uncertain which of the two contexts is correct decide which interpretation should be assigned?

Contribution tracking is an attempt to get a handle on these kinds of problems. The general approach is to take principled models of interpretation in context and situate them within specific *threads of interpretation* which an uncertain interlocutor can track over time. As an initial illustration of this approach, let us return to our example of Larry and Jeff’s dialogue (Section 2.4.3, page 29), in which Larry expresses his astonishment that Jason has come to “his place”, but Jeff is uncertain which place Larry means. We will use the following example as a first illustration of contribution tracking:

- Larry₁: You'll never believe it. Jason came to my place today!
 (Jeff is uncertain whether “my place” means Larry’s office or Larry’s house.)
- Jeff₂: Wow, ok.
 (Jeff remains uncertain whether “my place” means Larry’s office or Larry’s house.)
- Larry₃: Yeah. He was there when I arrived at 3pm.
 (Jeff now suspects that Larry must have meant Larry’s office, because Larry usually goes to his office in the afternoons. However, Jeff is unable to rule out that Larry meant Larry’s house.)
- (3.1) Jeff₄: I don’t believe it.
 (Though Jeff is uncertain exactly where Jason was, he decides to convey his incredulity that Jason came to whichever of Larry’s places he came to.)
- Larry₅: Yeah, I couldn’t believe it either when I pulled up and saw him sitting there on the front steps of my house.
 (Jeff now realizes Larry meant Larry’s house all along, and forgets about the interpretation involving Larry’s office.)

In this example, we have sketched, parenthetically, some of the reasoning involved in contribution tracking. Let us suppose that Jeff’s principled model of interpretation in context tells him that Larry’s expression *there* in Larry₃ is coreferential with Larry’s expression *my place* in Larry₁. However, suppose further that Jeff is unsure whether Larry’s expression *my place* in Larry₁ meant Larry’s office or Larry’s house. Our proposal is that, in order to continue participating in this dialogue, Jeff can spawn two threads of interpretation, a Larry’s office thread (t_{office}) and a Larry’s house thread (t_{house}). On the office thread, what Larry asserts — i.e. *contributes* — in Larry₃ is that Jason was at Larry’s office when Larry arrived at the office at 3pm. On the house thread, what Larry contributes in Larry₃ is that Jason was at Larry’s house when Larry arrived at the house at 3pm. Within each thread, Larry’s expression *there* in Larry₃ is coreferential with Larry’s expression *my place* in Larry₁. Jeff just isn’t sure which of these two threads captures the contributions that Larry has been trying to make to their dialogue.

By spawning two threads of interpretation, with different threads capturing Larry’s alternative contributions, Jeff can continue to assign coherent, principled interpretations to Larry’s ongoing utterances, and even participate in the dialogue himself. At Jeff₄, Jeff contributes his incredulity about the story Larry is telling, even though Jeff is uncertain about an aspect of this story that

affects the interpretation of his own contribution. Contribution tracking allows Jeff to formulate an utterance that makes a recognizable contribution *on either thread of interpretation*. Let us suppose that Jeff’s principled model of interpretation in context tells him that his own expression *it* in Jeff₄ is a discourse deictic pronoun which corefers with Larry₃’s sentence *He was there when I arrived at 3pm*. On thread t_{office} , Jeff can therefore anticipate that he would be contributing that he doesn’t believe that Jeff was at Larry’s office when Larry arrived at the office at 3pm. On thread t_{house} , Jeff can therefore anticipate that he would be contributing that he doesn’t believe that Jeff was at Larry’s house when Larry arrived at the house at 3pm. If Jeff is willing to make either of these contributions, it is easy to imagine that he would be willing to utter Jeff₄ despite his uncertainty about exactly which contribution he is making. (It may be more important to Jeff to play a participative role in Larry’s story telling than it is for him to know exactly what contribution he is making to that story telling.)

Later, in Larry₅, Larry finally reveals disambiguating information about his original meaning in saying *my place* in Larry₁. At this point, because Jeff has been tracking coherent threads of interpretation during this dialogue, he is able to retroactively eliminate t_{office} , and thereby collapse his uncertainty to the t_{house} interpretation of the entire dialogue.

In the vocabulary that we develop in the remainder of this dissertation, we will describe interlocutors, like Jeff here, as facing *transient uncertainties* which they can manage, and often eventually resolve, through a process of *contribution tracking*. Contribution tracking is a reasoning process that allows an interlocutor to use principled semantic and pragmatic models to attribute coherent contributions to speakers along alternative threads of interpretation. Crucially, contribution tracking allows interlocutors to exploit probabilistic reasoning to reallocate probability mass among the different threads of interpretation over time, while retaining a principled approach to linguistic interpretation in context within each thread. For example, though Jeff may have been strongly preferring the t_{office} thread after Larry₃, and during Jeff₄, the evidence Larry provides at Larry₅ may lead him to reallocate all the probability mass to t_{house} , and forget about t_{office} altogether.

We define and discuss contribution tracking more precisely in the next two sections, and then move on to a description of the detailed computational framework within which we have implemented it.

3.1.2 Definition of contribution tracking

We will say that a dialogue agent A implements contribution tracking if, and only if, both of the following conditions are satisfied:

1. A 's interpretation process for observed utterances (actions) is able to hypothesize alternative contributions:
 - (a) under uncertainty about the prior contributions that have been made by both interlocutors;
 - (b) considering the coherence of each hypothesized contribution in relation to the alternative prior contributions that may have been made; and
 - (c) assigning probabilities or relative confidences to alternative hypothesized contributions.

2. A 's generation process is able to identify contributive utterances (actions) to perform:
 - (a) under uncertainty about the prior contributions that have been made by both interlocutors;
 - (b) considering the coherence of each hypothesized contribution (for A 's utterance or action) in relation to the alternative prior contributions that may have been made; and
 - (c) assigning probabilities or relative confidences to alternative hypothesized contributions.

In this definition, we place a set of “boundary conditions” on the way an implemented dialogue agent reasons about the interpretations of utterances and tracks alternative interpretations over time. These boundary conditions are intended to characterize an agent's ability to pursue more than one thread of interpretation across several turns in dialogue, while placing minimal restrictions on the representations and algorithms that the agent uses to do so.

Requirements 1(a) and 2(a) require that the agent's interpretation and generation processes be able to function under (at least some kinds of) uncertainty about the prior contributions that the two interlocutors have made (perhaps after summarizing this uncertainty as uncertainty about the current dialogue state). This allows us to view the agent's interpretation and generation processes as functioning in a setting in which multiple threads of interpretation are viable. By requiring that these processes be sensitive to the prior contributions of *both* interlocutors, i.e. to the agent's own prior contributions as well as to the user's prior contributions, we enforce a minimal degree of consistency with mainstream linguistic theories of interpretation in context, which view the evolving context as a function of what the previous utterances of *both* interlocutors have meant.

Requirements 1(b) and 2(b) mean, essentially, that the agent must make some sort of effort to view linguistic interpretation as a function of the context that the prior contributions have created. This is a very weak requirement, from the standpoint of modern semantic and pragmatic theorizing. It does however rule out, for example, an agent which decides what interpretation should be assigned to its users' (or its own) utterances without any consideration at all of the current context or dialogue state. We thus intend contribution tracking to be a process of relating individual utterances to the evolving context, over time. This is also what allows us to view the agent as extending individual threads of interpretation as each new utterance or action occurs. The result is that we can view these threads of interpretation as chains of contextual reasoning that allow the agent to rationalize a subdialogue, despite a certain amount of uncertainty about what is happening in that subdialogue.

Requirements 1(c) and 2(c) require that the agent use some sort of probabilistic determination or ranking process that allows it to assess the relative likelihood of alternative threads of interpretation (perhaps summarizing this assessment as uncertainty about the next dialogue state). This is a big part of what makes contribution tracking a *tracking* problem, in which additional evidence is brought to bear, over time, in the agent's assessment of an evolving situation which is not directly observable.

Finally, condition 2 requires that the agent's generation process identify *contributive* utterances or actions to perform, in the sense of Chapter 2. This condition connects contribution tracking to a broadly collaborative view of language, in which interlocutors aim to make recognizable contributions to their ongoing tasks. In particular, it allows us to maintain this collaborative view of language even when an agent is actively pursuing several threads of interpretation for its dialogues.

Over all, our definition of contribution tracking describes a pragmatic reasoning process in which it may not be possible for contributions to be identified immediately — sometimes even by the actor that makes them. Rather, interlocutors select and perform *contributive* actions incrementally, with additional actions by one or both interlocutors helping to disambiguate a coherent thread of interpretation over time, and with both interlocutors working to track each other's (and their own) contributions as the dialogue evolves.

Partly as a reflection of the relative immaturity, in our view, of computational modeling of human-like reasoning about language use in context, our definition is remarkably noncommittal about detailed linguistic theories of human pragmatic reasoning; it draws on these theories only to a depth that yields a few basic boundary conditions on how the agent conceptualizes its interpretation processes. This definition nevertheless makes contribution tracking a substantially different reasoning process than is employed in existing computational approaches to tracking dialogue states under uncertainty. The closest computational approach is the use of POMDPs to model dialogue state

in spoken dialogue systems. We will now, therefore, relate POMDP approaches to contribution tracking in some detail.

3.1.3 Comparison to POMDP-based dialogue modeling

In this section, we discuss the relation between existing POMDP-based approaches to dialogue modeling and contribution tracking, as defined in Section 3.1.2. As discussed in Section 1.1 (page 4), a standard technique (Roy et al., 2000; Williams and Young, 2006, 2007) in POMDP-based dialogue systems is to simply identify the dialogue state with the user’s private state or goal. For example, in the human-robot dialogue application of Roy et al. (2000), the state, about which the POMDP-based dialogue system is uncertain, is the command the user is trying to give the robot. Similarly, in the POMDP-based call center application of Williams (2008), the state is the directory listing (name and listing type) that the human caller wishes to be connected to. In both cases, the system selects actions in a way that tries to manage or reduce the system’s uncertainty about the underlying user goal. The system’s uncertainty arises in the interpretation of ASR representations of spoken user utterances. In this setting, the POMDP approach supports robust strategies for optimizing a hand-crafted, task-specific utility model. This mostly amounts to satisfying the user’s goal without annoying them too much.

The easiest way to relate the POMDP approach to contribution tracking is by thinking of a “contribution”, in the POMDP framework, as the underlying state transition that occurs at each time step.¹ We thus essentially conceptualize a contribution as a change in the user’s private goal. With this perspective, the POMDP approach does yield uncertainty about the prior contributions that both interlocutors have made, in line with requirements 1(a) and 2(a), because the user’s private goal can sometimes change from one time step to the next. (In POMDP-based dialogue models, the user’s goal can change autonomously or in response to a system utterance or action.) With respect to requirements 1(b) and 2(b), we can also (sort of) think of the POMDP as considering the coherence of hypothesized contributions as the dialogue evolves. It does so by way of its observation and state transition models, $P(o|s, a)$ and $P(s'|s, a)$ respectively, which give the probability of observing user response o and, respectively, of moving to state s' , if the previous state was s and the system action was a . In a way, these models allow the system to reason about the “coherence”, understood in probabilistic terms, of specific state transitions as each system action a and user response o occur.

¹It may also seem tempting to identify a user contribution with the observation in a POMDP, and a system contribution with the action selected by the system’s policy. However, in the POMDP framework, both the observation and the system’s chosen action are known to the system with certainty. So there would be no uncertainty about either user or system contributions, which is contrary to requirements 1(a) and 2(a) in our definition.

POMDPs also satisfy 1(c) and 2(c) in their belief update step, which establishes a new belief $P(S_t)$ about the current user goal S_t at each time t .

The difference between a POMDP approach and contribution tracking, however, becomes clear when we consider whether the POMDP action policy — the POMDP-based agent’s “generation process” — selects an action that is *contributive*, as required by condition 2 in our definition of contribution tracking. As described in Chapter 2, acting contributively means acting with an expectation that your contribution will be recognized. Here we encounter some conceptual difficulties. First, if the state is identified with the user’s goal, and system contributions are understood as underlying state transitions, i.e. as specific changes in the user’s private goal, then what it means for the user to recognize the system’s contribution is for the user to recognize how their own goal has changed in response to an observed system action or utterance. But this is a bit odd — the user presumably knows their own goals, and knows when they change. (We could perhaps instead conceptualize a system contribution as an *intended* update to the user’s private goal, but because system actions are actually chosen to maximize a reward signal under uncertainty about which state transition will occur, there may not be any specific state transition that we could distinguish as the system’s intended one.)

At a deeper level, though, to identify the state with the user’s private goal or state seems to sacrifice any theoretical view of linguistic interpretation in dialogue as a coordination problem in which the two interlocutors aim to efficiently understand and be understood by *each other*. As discussed in Section 3.1.1 (page 62), this theoretical perspective pervades a wide range of work on linguistic interpretation in human-human dialogue. It is also one of the fundamental motivations for our definition of contributive action as a design goal for our dialogue agents: we want our agents to say and do things that their users will understand in context. By contrast, in current POMDP-based dialogue systems, there is generally no question of whether the user will understand the system’s utterances; rather, the overarching design goal is to build a system that can somehow decipher a user’s wishes in the face of ubiquitous errors in speech recognition.

Concretely, in selecting a system utterance, these systems generally draw from a small fixed inventory of utterances that have been hand-authored by a system designer to effectively convey a specific message to the user. In particular, the context-sensitivity of utterance meaning is not an explicit part of the modeling of an utterance as an action. This makes it difficult to see how a POMDP approach could robustly generate the various kinds of coordinated, context-sensitive linguistic expressions that pervade human-human dialogue, such as referring expressions, pronouns, and other kinds of anaphoric expressions. (A non-productive, “brute force” approach could be

employed in narrow application domains, however. On this approach, a training step would learn a policy that allows a system that is uncertain about the user’s goal to make a performance-optimizing selection from an exhaustive list of all the alternative context-dependent utterances in the domain.)

By contrast, the selection of system utterances and actions that the user will understand as coherent contributions in context is what contributive action and contribution tracking are all about. In particular, they aim to facilitate this selection process by making it possible for dialogue systems to organize the real-world uncertainties they encounter into coherent threads of context-sensitive interpretation, to track these threads using flexible probabilistic inference, and to make coherent, recognizable contributions of their own despite the uncertainty they are dealing with.

In the rest of the chapter, we present a general computational architecture (RUBRIC) which modularizes the detailed reasoning involved in contribution tracking, and an example dialogue application (COREF) that implements this reasoning.

3.2 Collaborative reference

3.2.1 Referring as a collaborative activity

The development of the COREF agent was originally inspired by empirical work on task-oriented human-human dialogue by Herb Clark and colleagues, especially Clark and Wilkes-Gibbs (1986) and Brennan and Clark (1996). Clark and Wilkes-Gibbs (1986) draw on observations of a collection of dialogue transcripts to argue against what they describe as the traditional *literary model* of definite reference, which assumes the following:

1. The reference is expressed linguistically with one of three standard types of noun phrase — a proper noun (for instance, *Napoleon*, *King George*), a definite description (*this year*, *the man with the moustache*), or a pronoun (*he*, *this*, *they*).
2. The speaker uses the noun phrase intending the addressee to be able to identify the referent uniquely against their common ground.
3. The speaker satisfies her intention simply by the issuing of that noun phrase.
4. The course of the process is controlled by the speaker alone.

They argue that these four claims are very coarse idealizations of the actual process of referring that occurs in human-human dialogue. For example, consider the following dialogue:²

²This and subsequent example dialogues in this section are discussed in Clark and Wilkes-Gibbs (1986), which also provides citations to original sources for some of these example dialogues.

- S_1 : Take the spout—the little one that looks like the end of an oil can—
 (3.2) J_2 : Okay.
 S_3 : —and put that on the opening in the other large tube. With the round top.

In S_1 , S does not satisfy her intention to refer to the spout by simply issuing a single definite noun phrase (assumptions 1, 3). Rather, S begins by uttering *the spout*, but apparently judges or senses that J will not be able to identify the intended referent. So S immediately initiates what Clark and Wilkes-Gibbs call an *expanded noun phrase* by inserting a second definite noun phrase: *the little one that looks like the end of an oil can*. The course of the referring process seems to depend on J 's response *Okay* in J_2 (contrary to assumption 4), after which S continues the pending utterance with S_3 . Similarly, after uttering *the other large tube* in S_3 , S apparently judges that description insufficient and adds *With the round top*.

Assumption 2 also idealizes a speaker's expectation that the hearer will be able to correctly identify each intended referent against the common ground:

- S_1 : Okay now, the small blue cap we talked about before?
 J_2 : Yeah.
 S_3 : Put that over the hole on the side of that tube—
 J_4 : Yeah.
 S_5 : —that is nearest to the top, or nearest to the red handle.

Here, S issues *the small blue cap we talked about before* with a rising intonation, as what Clark and Wilkes-Gibbs call a *trial noun phrase* that the hearer is invited to provide feedback on, as J does here in J_2 . Thus, a speaker may be substantially uncertain whether the addressee will be able to successfully interpret a referring expression that the speaker has decided to issue. Further, this uncertainty may lead to an interruption of the overall process of utterance generation. For example, in this dialogue, S could have simply uttered, *Put the small blue cap we talked about before over the hole on the side of that tube that is nearest to the top, or nearest to the red handle*, and only then invited feedback from J . Instead, S preferred to receive incremental feedback on specific referring expressions whose interpretation may have been problematic.

Sometimes a speaker is not just uncertain whether the addressee will be able to interpret a referring expression, but actually needs the addressee's help even to *construct* a referring expression:

A: The tree has, uh, uh...

B: Tentworms.

A: Yeah.

B: Yeah.

This issue of remembering or negotiating appropriate vocabulary to use in a particular description is more general, as illustrated in the following example from the object identification game studied in Brennan and Clark (1996):

Director: a docksider

Matcher: a what?

Director: um

Matcher: is that a kind of dog?

Director: no, it's a kind of um leather shoe, kinda preppy pennyloafer

Matcher: okay, okay, got it

In this example, the indefinite description *a docksider* results in a multi-utterance subdialogue in which one of the interlocutors (Matcher) actually learns a new vocabulary item before correctly identifying the original intended referent of *a docksider*.

Clark and Wilkes-Gibbs (1986) catalog a number of additional observations of this sort, in which successful reference involves the active participation of both the initiator of the reference and the recipient of the reference. When taken together, these phenomena suggest that reference in dialogue looks very much like a collaborative activity in which two interlocutors temporarily work together toward the goal of coming to an agreement about the identity of an object that one of them intends to single out to the other. Contrary to the literary model, this collaborative activity seems to have these features:

1. The reference is often *initiated* linguistically with one of three standard types of noun phrase — a proper noun (for instance, *Napoleon*, *King George*), a definite description (*this year*, *the man with the moustache*), or a pronoun (*he*, *this*, *they*). However, the initial noun phrase may trigger a multi-utterance dialogue involving linguistic expressions of heterogeneous form and function.
2. The speaker and the addressee work together to enable the addressee to be able to identify the referent uniquely.

3. The speaker satisfies her intention when the addressee successfully identifies the intended referent.
4. The course of the process is controlled by both the speaker and the addressee.

Our development of COREF was inspired by the challenges this perspective on reference seems to create for an implemented dialogue system. For example, in a standard NLG pipeline (Reiter and Dale, 2000), and in many implemented systems, the generation of referring expressions (GRE) is viewed as a distinct module or stage of processing. The output of the GRE process is expected to be incorporated into the system's overall output, for example by inserting the generated noun phrase into a larger sentence to be uttered by the system.

However, a collaborative view of reference suggests a fundamental orientation toward the multi-utterance *two-agent task* of referring rather than the specific *linguistic form* that a particular (attempted) referring expression takes. Drawing on the examples above, we can imagine that a human-like dialogue system that is trying to refer to an object might variously employ expanded noun phrases, trial noun phrases, and installment noun phrases; it might solicit online feedback and clarification questions from its user; and it might even exploit vocabulary provided by its user to help its referential task succeed.

It is possible to implement some of this broad functionality using a formal model that treats the product of the activity as a single evolving linguistic form, as in Heeman and Hirst (1995). For example, in a dialogue like (3.2) on page 71 above, the ultimate reference to the spout in S_1 could be summarized as *the little spout that looks like the end of an oil can*. However, in other human-human dialogue examples, it is more difficult to identify any single linguistic form as the product of the activity. For example, consider this dialogue (Clark and Wilkes-Gibbs, 1986):

- B_1 : How long y'gonna be here?
 A_2 : Uh- not too long. Uh just til uh Monday.
 B_3 : Til- oh yih mean like a week f'm tomorrow.
 A_4 : Yah
 B_5 : (continues)

Here, one might judge that the ultimate product of the reference begun in A_2 is the single word *Monday*, or the clarified indefinite description *a week f'm tomorrow*, or even the appositive construction *Monday, a week f'm tomorrow*.

Our view is that in order to replicate the flexibility that human speakers exhibit in collaborative reference subdialogues, we need to be able to explain, at a computational level, each individual linguistic form they use in terms of an underlying task-oriented collaboration. In light of the various phenomena that arise in these subdialogues, it seems unmotivated to require that these explanations somehow coerce all the linguistic forms previously used by the interlocutors into a single summary expression. Instead, we prefer to view any single summarizing expression that might be formulated as an abstract by-product of the underlying reference task, rather than the basic purpose or product of the task.

To implement this approach, however, we need to model formally the two-agent task of reference, including the types of dialogue acts it involves, and to articulate a practical strategy for completing this task collaboratively using individual utterances. We have implemented this approach in the COREF agent, targeting dialogues in an object identification game adapted from the experiments of Clark and Wilkes-Gibbs (1986) and Brennan and Clark (1996), but with an eye toward the eventual objective of making human-like collaborative reference abilities more widely available in human-machine dialogue systems. We turn now to the details of COREF’s object identification game.

3.2.2 COREF’s object identification game

COREF participates in a two-agent object identification game which we adapted from the experiments of Clark and Wilkes-Gibbs (1986) and Brennan and Clark (1996). The game plays out in a special-purpose graphical user interface which we have developed. The interface, which can support either human-human or human-agent interactions, is shown in Figure 3.1.

The general objective of the game is for the two agents to work together to create a specific configuration of objects, or a “scene”, by adding objects into the scene one at a time. In all the games we discuss in this dissertation, the scene takes the form of a 2x2, 3x3, or 4x4 grid of objects.

The experimental design used by Clark and Wilkes-Gibbs (1986) and Brennan and Clark (1996) facilitated the study of purely verbal communication: the two players were given identical sets of objects (on flash cards which had been shuffled), and were separated by a physical barrier in such a way that they could hear each other speak but could not see each other. This ruled out the use of pointing gestures or communication through other kinds of non-verbal behavior.

We employ a conceptually similar game design. In our case, the two players each participate using their own version of the interface. They participate from physically separated locations so

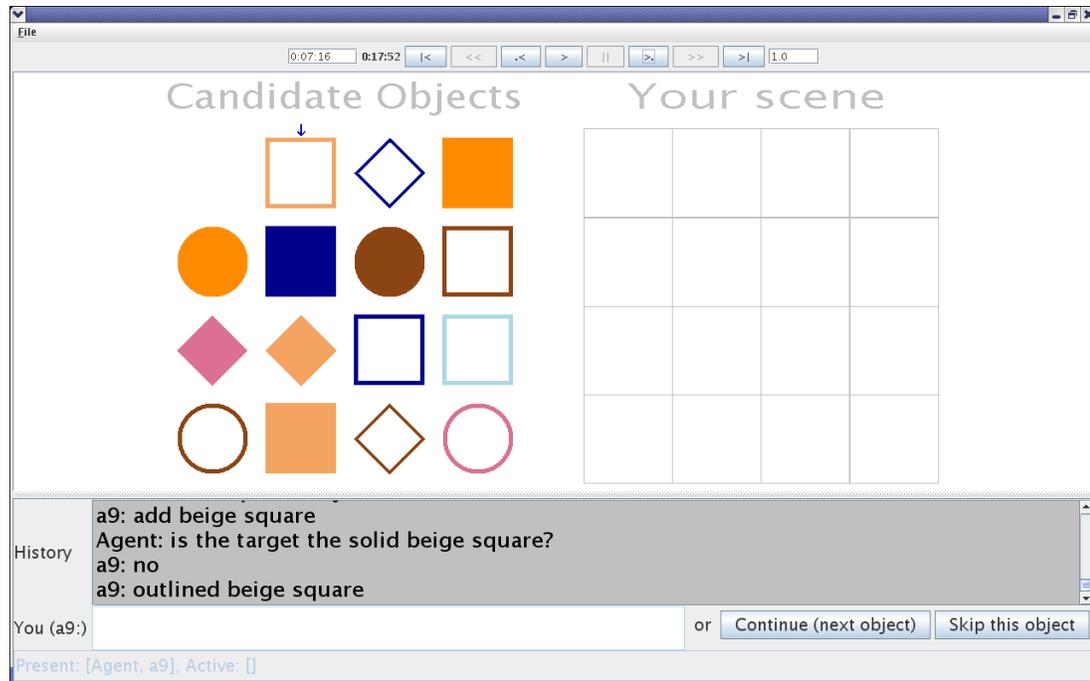


Figure 3.1: A human user plays an object identification game with COREF. The figure shows the perspective of the user (denoted a9). The user is playing the role of director, and trying to identify the square at upper left (indicated to the user by the blue arrow) to COREF. COREF’s perspective at the same point in time is shown in Figure 3.2.

that communication can only occur through the interface. Their interfaces display the same set of candidate objects to the two players, but the spatial locations of the objects are shuffled for each player. The shuffling undermines the use of spatial expressions such as “the object at the top left”. Figures 3.1 and 3.2 illustrate this arrangement.³

As in the experiments of Clark and Wilkes-Gibbs (1986) and Brennan and Clark (1996), one of the players, who plays the role of *director*, instructs the other player, who plays the role of *matcher*, which object is to be added next to the scene. As the game proceeds, the next target object is automatically determined by the interface and privately indicated to the director. The target object is indicated to the director using a blue arrow, as shown in Figure 3.1. (Note that the corresponding matcher’s perspective, shown in Figure 3.2, does not include the blue arrow.) The director’s job is then to get the matcher to click on (their version of) this target object.

³Note that in a human-human game, there are literally two versions of the graphical interface, on the separate computers the human participants are using. In a human-agent interaction, the agent does not literally use the graphical interface, but the information that the agent is provided is limited to the information the graphical interface would provide to a human participant. For example, the agent is not aware of the spatial locations of objects on the screen of its human partner.

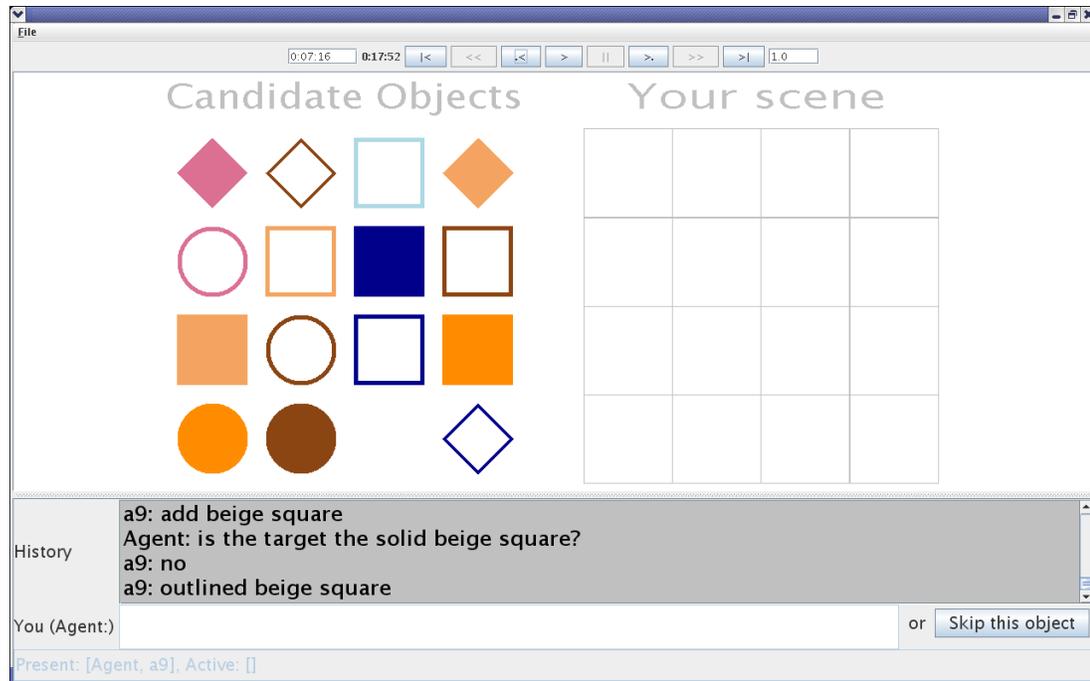


Figure 3.2: COREF plays an object identification game with a human user. The figure shows the perspective of COREF (denoted **Agent**). COREF is playing the role of matcher, and trying to determine which object the user wants COREF to identify. The user’s perspective at the same point in time is shown in Figure 3.1.

To achieve agreement about the target object, the two players can talk back and forth using written English, in an instant-messaging style modality, as illustrated in Figures 3.1 and 3.2. The interface provides a text box within which utterances can be typed. A typed utterance is only transmitted when the **Enter** key is pressed. Each player’s interface provides a real-time indication that their partner is “Active” while their partner is composing an utterance in their text box, but the interface does not show in real-time what characters are being typed. Thus, it is not possible for a player to view or interpret an utterance by their partner incrementally (as it is typed, word by word). Once the **Enter** key is pressed, the utterance appears on the screens of both players as part of a scrollable history field in the interface, which provides full access to all the previous utterances in the dialogue.

When the matcher clicks on an object they believe is the target, their version of that object is privately moved into their scene. The director has no visible indication that the matcher has clicked on an object. However, the director needs to click the **Continue (next object)** button (see Figure 3.1) in order to move the current target into the *director’s* scene, and move on to the next target

object. This means that the players need to discuss not just what the target object is, but also whether the matcher has added it into their scene, so that they can coordinate on the right moment to move on to the next object. If this coordination succeeds perfectly, then after the director and matcher have completed a series of objects, they will have created the exact same scene in their separate interfaces.

As the game proceeds, the director and matcher work through the creation of several different scenes. In all the experiments we discuss in this dissertation, the two players first work through 4 objects to create a 2x2 grid scene, then work through 9 more objects to create a 3x3 grid scene, and finally work through 16 more objects to create a 4x4 grid scene. Thus, there are a total of 29 target objects.

Both the director and matcher have the option to click a `Skip this object` button at any time. COREF never elects to skip an object, however. Human subjects are instructed that they have the option to press the `Skip this object` button if they feel they are no longer making progress on that particular object, but are asked not to skip an object just because it is taking a little longer than they would like. Human participants are also told that it is very important to be accurate, but that they should move right long, as they are being timed. The complete task instructions are given in Appendix A on page 210.

In the remainder of this dissertation, we will examine in detail many of the dialogue phenomena that arise in this object identification game.

3.3 The RUBRIC and COREF system architectures

In this section, we provide a high-level tour of COREF's architecture and implementation. We view COREF's implementation as an instantiation of a more general agent architecture which we call the RUBRIC (Robust Uncertain Basically Reversible Intentional Communication) architecture. The RUBRIC architecture is an attempt to specify a relatively generic, modular design and control flow for contribution tracking in an implemented dialogue agent. We begin in Section 3.3.1 with a discussion of the contributions we believe this architecture makes to dialogue system-building in general. In Section 3.3.2, we give an example COREF dialogue which we will use to illustrate various aspects of COREF's design. We then proceed, in subsequent sections, through a discussion of the various modules that are part of the RUBRIC architecture. In each case, we define the problem-solving the module is expected to perform, and sketch COREF's implementation of this problem-solving.

For the interested reader, COREF is implemented in the Java programming language. The control flow for the RUBRIC architecture, which we define in Section 3.3.3, is also implemented in Java. The individual RUBRIC modules are defined as Java interfaces.

3.3.1 Contributions to system architectures for dialogue systems

The RUBRIC architecture is an attempt to specify a relatively generic, modular design and control flow for contribution tracking in an implemented dialogue agent. As will become clear as we present the architecture in the following sections, many aspects of this architecture are borrowed from, or otherwise shared with, the modular architectures of various other dialogue systems. However, there are three aspects to the RUBRIC design which we believe constitute contributions which may be usefully exploited in the design of subsequent dialogue systems. We present these contributions here, before delving into the details of the architecture.

The first contribution relates to the division between engineering and “deep coherence” approaches to dialogue systems, which we discussed in Section 1.1 (page 4). In that discussion, we presented the dialogue systems community as divided into an engineering camp, which aims for high task performance using mostly data-based techniques, and a deep coherence camp, which aims for high task performance using mostly hand-built, rule-based models based on theories of coordination in dialogue.

From this perspective, the architecture we present here articulates a new third option: the pursuit of *contribution tracking*. Our architecture shares with the deep coherence camp a commitment to a clear theoretical foundation for coordination in dialogue, and shares with the engineering camp a commitment to clear probabilistic models that support data-based techniques for scaling up a dialogue system’s abilities. The intersection of these commitments is a new focus on applying probabilistic modeling to the fundamental mechanisms of coordination in language use. This differs from the probabilistic modeling for shallow task optimization typically used in the engineering camp. Our focus lies in *understanding* and *being understood* by the system’s human user, and in maintaining a coherent rationalization of extended subdialogues. This creates differences both in the methodological target of our probabilistic models and in the data set needed to tune those models. (See Chapter 4.) While we believe our architecture could be used to construct a contribution tracking agent that optimizes its dialogue policy for task performance, we have not, to date, pursued this kind of policy optimization in COREF. Instead, COREF relies on a hand-crafted dialogue policy, which we discuss further below.

The second architecture contribution is an approach that largely unifies the treatment of verbal utterances and non-verbal motor action. As we will show in our detailed discussion of COREF, it is possible, within the framework developed here, to view non-verbal motor action as performing many of the same communicative functions that are typically associated with verbal utterances. For example, a motor action in COREF’s domain can straightforwardly contribute that the previous utterance has been understood, or carry other kinds of implicatures. This unification of pragmatic inference about non-verbal and verbal action derives from the parallelism in our definitions of contributive action and language use in Chapter 2. While the communicative potential of non-verbal action has been studied by many researchers, we believe this architecture provides a particularly clear unification of the interpretation of verbal and non-verbal action in dialogue.

The third contribution of our architecture is its emphasis on the reversibility of representations and reasoning in dialogue. Unlike many dialogue systems, COREF was designed “from the ground up” to exploit reversible representations and reasoning wherever possible. For example, it is common for an implemented system’s NLU and NLG modules to diverge significantly in terms of their internal representations and the problem-solving performed (Reiter and Dale, 2000, p. 2-3). One common source of this divergence is the asymmetry of ASR errors, which are a major factor in NLU, but not in NLG. More generally, resolving uncertainty is often seen as fundamental to NLU but not to NLG.

COREF, in contrast, is an example of a dialogue system in which there is minimal divergence between NLU and NLG. The two modules exploit the same grammar and other knowledge resources for interpretation.⁴ Thus, unlike many dialogue systems, COREF can understand anything it can say, and it can (in principle) say anything it can understand.⁵ Further, the problem-solving COREF performs in interpretation and generation is parallel: the two modules rely on the same resources, and they both construct and output intention representations. In fact, this parallelism is enforced as a basic aspect of the design: COREF relies on its NLU module to interpret its *own* utterances, and to update the dialogue state appropriately, after it has uttered them (rather than letting its NLG module report what the utterance meant). A similar parallelism applies to the generation and interpretation of non-verbal motor actions.

Finally, COREF was designed from the ground up to be able to play any role in any task it can participate in. Thus, it can play either director or matcher in a `CollabRef` task, it can both ask and answer questions, etc. COREF, and its architecture, therefore, present an interesting case study of

⁴The use of a teletype interface, which sidesteps the issue of ASR error, is probably crucial to allowing NLU and NLG to use the same grammar.

⁵There are various utterances that COREF can understand, but which it would never actually say, due to its particular policies for deciding what to say.

the viability of reversible representations and reasoning in dialogue systems more generally.

3.3.2 An illustrative COREF subdialogue

We will use the example subdialogue in Figure 3.3 to illustrate our discussion of the RUBRIC and COREF system architectures. The figure depicts COREF’s pragmatic reasoning during a short subdialogue about a single target object. COREF plays the role of director, and starts by saying to s1, a human experimental subject who plays the role of matcher, *the blue circle*.

The figure organizes the ensuing dialogue into a series of utterance-by-utterance reasoning steps. In the topmost row, COREF’s perspective on the visual objects is given under the heading ‘EI’ (for ‘Experiment Interface’). An ellipsis indicates that COREF’s version of the experiment interface has not changed from one step to the next.

The second row indicates the “time”. Time is counted in sensory events, which COREF either experiences (as in the case of a user utterance) or else causes s1 to experience (as in the case of an utterance by COREF). The sensory event (‘SE’) is indicated in the third row.

The fourth row depicts COREF’s contribution tracking (‘CT’). This kind of diagram depicts contribution tracking in terms of evolving dialogue states – with state identifiers like ‘s1934’, ‘s2106’, etc. – that are connected by the interpretations that COREF assigns to sensory events. For example, COREF assigns interpretation $i_{1,1}$ to its own utterance of *the blue circle*, and views this interpretation as causing a transition from state s1934 to state s2106. The form of these interpretations, and how COREF identifies them, will be discussed further below.

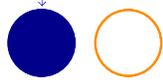
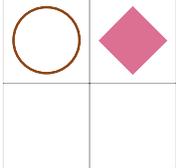
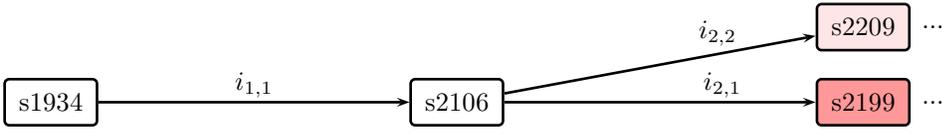
Finally, COREF assigns probabilities to the different dialogue states it believes it might be in. The probabilities that COREF assigns, at each point in time, are indicated by coloring the corresponding state nodes with shades of red. A completely white node, such as s1934 in the figure, indicates that COREF is completely certain, assigning probability 1 to that state. When COREF is uncertain, shades of red are used to indicate the relative probability assigned to each state, with darker shades of red indicating lower probability being assigned to the state.⁶ For example, the figure shows that at times 3, 4, and 5 COREF is uncertain which of two dialogue states is the correct one. A node depicted with a crosshatch pattern, such as s2704 at time 6, indicates a state that COREF has made a strategic decision to “drop”, or stop tracking.

In the following sections, we will use this short subdialogue to provide concrete examples of the detailed reasoning performed by the various internal modules that realize COREF’s pragmatic

⁶If you are reading a version of this document that has been printed in gray scale, darker shades of gray indicate lower probability being assigned to the state.

reasoning. An important theme will be that, throughout this subdialogue, COREF faces considerable uncertainty about all of the following: what contributions its human interlocutor is making with their utterances, what dialogue state they are in, and what contributions COREF itself is making with *its* utterances. Despite this uncertainty, COREF is able to speak contributively and complete the object successfully.

Figure 3.3: An illustrative COREF subdialogue.

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div> <p>...</p>		
Time:	1	2	
SE:	COREF: <i>the blue circle</i>	s1: <i>ok</i>	
CT:	 <pre> i_{1,1} = ⟨ s1 : tacitNop[[COREF does clickContinue[]]], COREF : pushCollabRef[COREF, s1, t3], COREF : addcr[t3, equal(t3, e2_2)], COREF : setPrag[inFocus(Z), inFocus(t3)] ⟩ i_{2,1} = ⟨ s1 : tacitNop[[COREF does say[the blue circle]]], s1 : setVarValue[t3, e2_2], s1 : addToScene[e2_2], s1 : past[s1, addToScene[e2_2]] ⟩ i_{2,2} = ⟨ s1 : nop[[COREF does say[the blue circle]]] ⟩ </pre>		
EI:	
Time:	3	4	
SE:	COREF: <i>did you add it ?</i>	s1: <i>yes</i>	

continues on next page...

Figure 3.3: An illustrative COREF subdialogue. (continued)

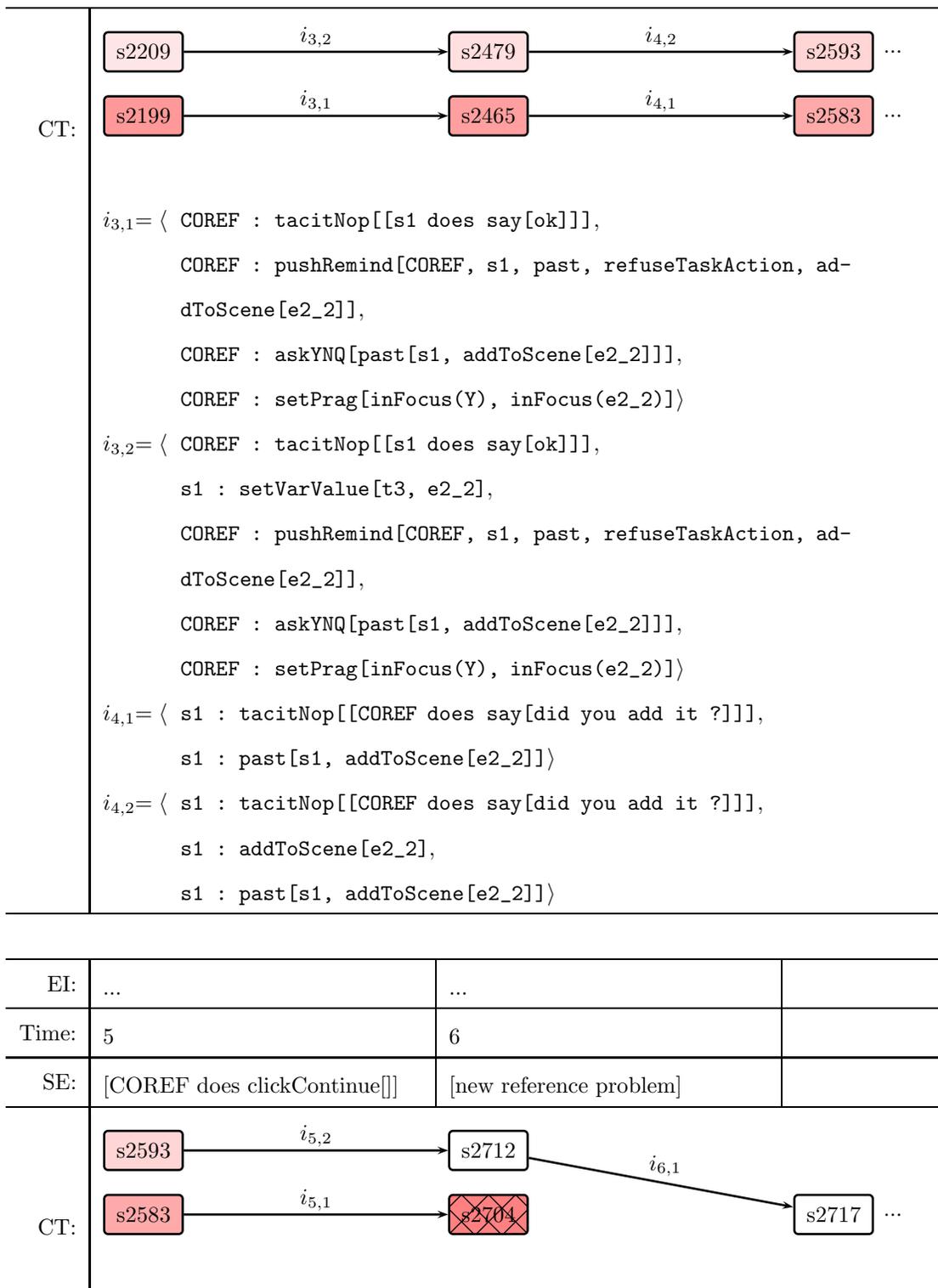
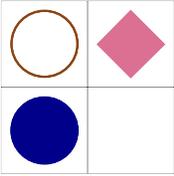
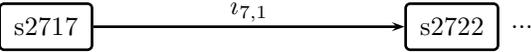
*continues on next page...*

Figure 3.3: An illustrative COREF subdialogue. (continued)

	$i_{5,1} = \langle \text{COREF : tacitNop}[[s1 \text{ does say}[\text{yes}]]],$ $\text{COREF : continueTask}[\text{t3}] \rangle$ $i_{5,2} = \langle \text{COREF : tacitNop}[[s1 \text{ does say}[\text{yes}]]],$ $\text{COREF : continueTask}[\text{t3}] \rangle$ $i_{6,1} = \langle \text{COREF : perceive}[\text{PerceivedNewReferenceProblemEvent}\langle \text{t4} \rangle] \rangle$		
EI:	Candidate Objects 	Your scene 	
Time:	7		
SE:	[perceived 4 figure objects]		
CT:	 $i_{7,1} = \langle \text{COREF : perceive}[\text{PerceivedFigureObjectsEvent}\langle [\text{e3}_3, \text{e2}_2, \text{rh1}_1, \text{e0}_0] \rangle] \rangle$		

3.3.3 Control flow

In this section, we survey the high-level flow of control in a RUBRIC agent. We discuss the problem-solving performed by the individual modules in more detail in subsequent sections.

The flow of control in a RUBRIC agent is specified in Figure 3.4. The agent a maintains its uncertainty about which dialogue state S_t it is in at time t , given its available historical evidence h_t , as a probability distribution $P(S_t|h_t)$. We'll describe a dialogue state s as *viable* (from a 's perspective), at time t , iff $P(S_t = s|h_t) > 0$. The *viable states* v_t at time t are thus:

$$v_t = \{s \mid P(S_t = s|h_t) > 0\}.$$

(In this and subsequent sections, we will use uppercase names like ' S_t ' for variables about whose

RUBRIC agent

Problem CONTRIBUTE($a, u, s_0; \mathbf{S}, \text{SEI}, \text{UC}, \mathbf{G}, \text{IG}, \text{MA}$)

Description: RUBRIC agent a contributes to a task with user u .

Input (task): Agent identifier a , user identifier u , a 's initial dialogue state s_0 .

Input (resources): Sensorium \mathbf{S} , sensory event interpreter SEI, update control UC, grammar \mathbf{G} , intention generator IG, motor apparatus MA.

Algorithm:

$h_0 \leftarrow \langle \rangle$, $P(S_0 = s_0 | h_0) \leftarrow 1$, $t \leftarrow 0$, generate \leftarrow false

while termination condition not met:

 if a new sensory observation is available:

$o \leftarrow \mathbf{S}.\text{SENSE}()$

$P(E|o, S_t) \leftarrow \mathbf{S}.\text{PERCEIVE}(o, P(S_t|h_t))$

$P(I|E, o, S_t) \leftarrow \text{SEI}.\text{INTERPRET}(a, o, P(S_t|h_t), P(E|o, S_t); \mathbf{G}, \text{UC})$

$h_{t+1} \leftarrow \text{APPEND}(h_t, a, o)$

$P(S_{t+1}|h_{t+1}) \leftarrow \text{UC}.\text{FILTER}(a, o, P(S_t|h_t), P(E|o, S_t), P(I|E, o, S_t))$

$t \leftarrow t + 1$, generate \leftarrow true

 else if generate:

 if the agent is willing to generate now:

$\langle m^*, o_u, P_u(E|o_u, S_t), P_u(I|E, o_u, S_t) \rangle \leftarrow \text{IG}.\text{GENERATE}(P(S_t|h_t), a, u; \mathbf{G}, \text{UC}, \text{SEI})$

 if agent is willing to act now:

$\text{MA}.\text{DO}(m^*)$

$h_{t+1} \leftarrow \text{APPEND}(h_t, u, o_u)$

$P(S_{t+1}|h_{t+1}) \leftarrow \text{UC}.\text{FILTER}(u, o_u, P(S_t|h_t), P_u(E|o_u, S_t), P_u(I|E, o_u, S_t))$

$t \leftarrow t + 1$, generate \leftarrow false

 else if pulse timeout reached:

 generate \leftarrow true

Figure 3.4: The flow of control in a RUBRIC agent.

values the agent is generally uncertain, and lowercase names like ‘ h_t ’ and ‘ v_t ’ for information and representations the agent has direct access to and does not face uncertainty about.)

A RUBRIC agent interprets its sensory observations in several steps. First, the agent has a module we call the *Sensorium* (**S**) which outputs raw sensory events:

$$o \leftarrow \mathbf{S.SENSE}()$$

A sensory event o might be an audio sample from a microphone, or a camera image, for example. The Sensorium module is also responsible for probabilistically classifying each raw sensory event o as some perceptual event E :

$$P(E|o, S_t) \leftarrow \mathbf{S.PERCEIVE}(o, P(S_t|h_t))$$

The perceptual event E might be the speech identified by an automatic speech recognition algorithm, or a user motor action detected in a camera image, for example. The output $P(E|o, S_t)$ of this perceptual classification specifies the conditional probability $P(E = e_j|o, S_t = s_k)$ of one or more perceptual events e_j for each viable state $s_k \in v_t$ in which o may have occurred.⁷

The final stage of interpretation is performed by a module we call the *Sensory Event Interpreter* (**SEI**), which probabilistically understands the observation o and perceptual event E as some interpretation I :

$$P(I|E, o, S_t) \leftarrow \mathbf{SEI.INTERPRET}(a, o, P(S_t|h_t), P(E|o, S_t); \mathbf{G}, \mathbf{UC})$$

The interpretation I might be a sequence of dialogue acts or user actions that (must) have been performed, for example. The interpretation step identifies alternative interpretations given the agent’s uncertainty $P(S_t|h_t)$ and its perceptual classification $P(E|o, S_t)$, and using as additional resources a *grammar* **G** and an *update control* **UC**. We discuss the roles these resources play in interpretation below. The output $P(I|E, o, S_t)$ of interpretation specifies the conditional probability $P(I = i_l|E = e_j, o, S = s_k)$ of one or more interpretations i_l for each perceptual analysis e_j of o in each viable state $s_k \in v_t$.

After interpretation, the agent’s historical evidence is updated to include a ’s new sensory event

⁷Providing the distribution $P(S_t|h_t)$ as an input to **S.PERCEIVE** allows the perceptual classifier to identify the viable states.

o :

$$h_{t+1} \leftarrow \text{APPEND}(h_t, a, o)$$

A module we call *Update Control* (UC) is then responsible for updating the agent’s uncertainty about the new dialogue state:

$$P(S_{t+1}|h_{t+1}) \leftarrow \text{UC.FILTER}(a, o, P(S_t|h_t), P(E|o, S_t), P(I|E, o, S_t))$$

After this update, the agent’s clock advances and the agent becomes disposed to generate some actions of its own.

However, there may be times at which a RUBRIC agent is not willing to generate any actions to perform. For example, if the user is currently speaking, the agent might choose not to generate any actions of its own until the user stops speaking. If the generation condition is met, a module we call the *Intention Generator* (IG) module is consulted. The Intention Generator selects an action sequence m^* while anticipating how the user u will observe (o_u), perceive (E), and interpret (I) the agent’s performance of m^* :

$$\langle m^*, o_u, P_u(E|o_u, S_t), P_u(I|E, o_u, S_t) \rangle \leftarrow \text{IG.GENERATE}(P(S_t|h_t), a, u; \mathbf{G}, \text{UC}, \text{SEI})$$

In particular, generation starts from the agent a ’s uncertainty $P(S_t|h_t)$ about which dialogue state it is in. Given that uncertainty, the generation algorithm identifies a motor action sequence m^* for the agent to perform, anticipating that the user u will observe the agent’s performance of m^* with observation o_u , which u will perceive as $P_u(E|o_u, S_t)$ and finally interpret as $P_u(I|E, o_u, S_t)$. An important design feature is that these representations are parallel to those the agent constructs in interpreting its own sensory observations. The Intention Generator draws on several resources and modules to help it identify an acceptable output m^* , including the Sensory Event Interpreter. We discuss the roles these resources play in generation below.

After generation, provided the agent is willing to act (for example, the agent might decide not to act if the user has just started speaking), the agent instructs its Motor Apparatus module (MA) to execute the generated action sequence m^* :

$$\text{MA.DO}(m^*)$$

The agent’s historical evidence about the dialogue state is then updated to include the agent’s

estimate of the sensory event o_u that m^* has provided to the user u :

$$h_{t+1} \leftarrow \text{APPEND}(h_t, u, o_u)$$

In practice, of course, the agent can only estimate the actual observation o_u that the user will make. A simplifying assumption here is that the agent does not face uncertainty about o_u . For example, if the agent has generated a dialogue act, and its Motor Apparatus has executed the act by playing some speech-containing audio sample A to the user, the agent might assume the user will observe exactly this same audio wave form, and therefore simply let $o_u = A$.

Finally, the agent updates its uncertainty to reflect the interpretations it anticipates the *user* will assign to the actions the agent has performed:

$$P(S_{t+1}|h_{t+1}) \leftarrow \text{UC.FILTER}(u, o_u, P(S_t|h_t), P_u(E|o_u, S_t), P_u(I|E, o_u, S_t))$$

The agent then advances its clock and cancels its (now fulfilled) desire to generate.

3.3.3.1 COREF's control flow

COREF implements the RUBRIC control flow with one major simplification. Because COREF's object identification game uses a teletype interface, and because we provide COREF with direct access to the visual objects in its interface, and their properties, COREF does not face any uncertainty at RUBRIC's perceptual level. In terms of the RUBRIC control flow, we therefore unify each sensory observation o that COREF makes with its corresponding perceptual classification E , letting $E = o$. COREF's problem-solving in interpretation, generation, and filtering is correspondingly simplified. We summarize these simplifications in Figure 3.5.

The result is that, in COREF, all uncertainty arises at the level of interpretation and generation; no uncertainty arises in perception. Perceptual uncertainty is nevertheless modeled as part of the RUBRIC architecture, because many other dialogue agents do face uncertainty at the perceptual level (for example, arising in the output of speech recognition). We intend for the general RUBRIC architecture to clarify the reasoning such agents could use to implement contribution tracking under perceptual uncertainty.

COREF's decision-making about when it is willing to generate and to act is based on a simple turn-taking model in which the user is able to take and relinquish "the floor" at will. The user has the floor while they are typing an utterance. In particular, the user takes the floor by starting to type

COREF agent

Problem CONTRIBUTE($a, u, s_0; \mathbf{S}, \mathbf{SEI}, \mathbf{UC}, \mathbf{G}, \mathbf{IG}, \mathbf{MA}$)

Description: RUBRIC agent a contributes to a task with user u .

Input (task): Agent identifier a , user identifier u , a 's initial dialogue state s_0 .

Input (resources): Sensorium \mathbf{S} , sensory event interpreter \mathbf{SEI} , update control \mathbf{UC} , grammar \mathbf{G} , intention generator \mathbf{IG} , motor apparatus \mathbf{MA} .

Algorithm:

$h_0 \leftarrow \langle \rangle$, $P(S_0 = s_0 | h_0) \leftarrow 1$, $t \leftarrow 0$, generate \leftarrow false

while termination condition not met:

 if a new sensory observation is available:

$o \leftarrow \mathbf{S.SENSE}()$

$P(I|o, S_t) \leftarrow \mathbf{SEI.INTERPRET}(a, o, P(S_t|h_t); \mathbf{G}, \mathbf{UC})$

$h_{t+1} \leftarrow \mathbf{APPEND}(h_t, a, o)$

$P(S_{t+1}|h_{t+1}) \leftarrow \mathbf{UC.FILTER}(a, o, P(S_t|h_t), P(I|o, S_t))$

$t \leftarrow t + 1$, generate \leftarrow true

 else if generate:

 if user does not have the floor:

$\langle m^*, o_u, P_u(I|o_u, S_t) \rangle \leftarrow \mathbf{IG.GENERATE}(P(S_t|h_t), a, u; \mathbf{G}, \mathbf{UC}, \mathbf{SEI})$

 if user still does not have the floor:

$\mathbf{MA.DO}(m^*)$

$h_{t+1} \leftarrow \mathbf{APPEND}(h_t, u, o_u)$

$P(S_{t+1}|h_{t+1}) \leftarrow \mathbf{UC.FILTER}(u, o_u, P(S_t|h_t), P_u(I|o_u, S_t))$

$t \leftarrow t + 1$, generate \leftarrow false

 else if pulse timeout reached:

 generate \leftarrow true

Figure 3.5: The flow of control in the COREF agent.

an utterance, and relinquishes the floor either by pressing Enter (thus transmitting the utterance to COREF) or by pressing backspace until all the characters they have typed have been erased. COREF will only generate or perform actions when the user does not have the floor. This means that the user sometimes takes the floor while COREF is generating, and still has the floor when COREF finishes generating; in these cases, COREF abandons its generatum.

3.3.4 Dialogue state

The RUBRIC architecture makes no assumptions about the information present in an agent’s dialogue state. In general, the architecture is mostly consistent with the *information state approach* to dialogue management (Larsson and Traum, 2000). The primary implication of the information state approach is that a clean separation is made, at the computational level, between the agent’s dialogue state, the mechanisms by which the dialogue state is updated, and the agent’s control flow. The RUBRIC architecture makes these distinctions clearly; see the dialogue state S_t , update control UC, and CONTRIBUTE algorithm in Figure 3.4. However, RUBRIC generalizes the update mechanism from one in which a single dialogue state is deterministically updated, at each time step, to one in which the agent’s uncertainty $P(S_t|h_t)$ is *filtered* at each time step (DeVault and Stone, 2007).

3.3.4.1 COREF’s dialogue state

In this section, we give a detailed presentation of COREF’s dialogue state representation. We will show that this representation allows COREF to treat its object identification game as a two-agent dialogue task in the sense of Section 2.6.2. This will make it possible for us to characterize COREF’s motor actions and language use as contributive in subsequent sections. With this end in mind, we now present the various models COREF employs in order to represent its object identification game in terms of various subtasks, individual agent actions, and other components.

COREF’s global dialogue state (or “context”) at time t takes the form $c_t = \langle R, P, T, C, U, O, H \rangle$, where R is a set of objects yet to be identified, P is a set of propositions that have previously been added to the context (e.g. by a player action or utterance), T is a stack of active tasks, C is a set of constraint networks (one for each target object), U is the universe of discourse (a set of properties and objects), O is the set of visual objects present in the interface, and H is a historic record of previous events in the dialogue. As a concrete example, Figure 3.6 shows dialogue state s2106, which is the dialogue state COREF believes it is in at time 2 in the subdialogue given in Figure 3.3 (page 81).

<i>R</i>	{e2_2, e3_2}																
<i>P</i>	{director(COREF), matcher(s1), equal(t1, e0_0), varValue(t1, e0_0), inScene(e0_0, s1), finished(t1), equal(t2, rh1_1), varValue(t2, rh1_1), inScene(rh1_1, s1), finished(t2), currentTarget(t3), equal(t3, e2_2), inFocus(t3), ...}																
<i>T</i>	<table border="1"> <tr> <td>ManageAmbiguity(s1, [COREF does say[the blue circle]])</td> </tr> <tr> <td>CollabRef(COREF, s1, t3)</td> </tr> <tr> <td>CreateParallelScenes()</td> </tr> </table>	ManageAmbiguity(s1, [COREF does say[the blue circle]])	CollabRef(COREF, s1, t3)	CreateParallelScenes()													
ManageAmbiguity(s1, [COREF does say[the blue circle]])																	
CollabRef(COREF, s1, t3)																	
CreateParallelScenes()																	
<i>C</i>	<table border="1"> <tr> <td>director =</td> <td>COREF</td> <td>matcher =</td> <td>s1</td> </tr> <tr> <td>target =</td> <td>t3</td> <td>intended referent =</td> <td>?</td> </tr> <tr> <td>candidate referents =</td> <td>{e2_2, e3_2}</td> <td>positive constraints =</td> <td>{equal(t3, e2_2)}</td> </tr> <tr> <td>negative constraints =</td> <td>{}</td> <td></td> <td></td> </tr> </table> <p>(+ two omitted constraint networks for previous target objects t1 and t2)</p>	director =	COREF	matcher =	s1	target =	t3	intended referent =	?	candidate referents =	{e2_2, e3_2}	positive constraints =	{equal(t3, e2_2)}	negative constraints =	{}		
director =	COREF	matcher =	s1														
target =	t3	intended referent =	?														
candidate referents =	{e2_2, e3_2}	positive constraints =	{equal(t3, e2_2)}														
negative constraints =	{}																
<i>U</i>	{means, meanPredicateSymbol, meanEntity, nextCell, noBlocksLeft, sceneIsComplete, different, equal, crSpeaker, crHearer, varValue, finishedCreateParallelScenes, director, matcher, inScene, finished, recognizesIntention, ..., currentTarget, anyTarget, anyReferent, inTargetDomain, figureObjectColor, figureObject, squareFigureObject, rectangleFigureObject, rhombusFigureObject, ..., blackFigureObject, blueFigureObject, darkblueFigureObject, lightblueFigureObject, ..., recentlyIntendedConcretePredicate, recentlyIntendedNonPredicate, inFocus, COREF, s1, t1, e0_0, rh1_0, e2_0, e3_0, ...}																
<i>O</i>	{e2_2, e3_2, rh1_1, e0_0}																
<i>H</i>	(history omitted for space reasons)																

Figure 3.6: An example COREF dialogue state: s2106.

In the remainder of Section 3.3.4.1, we present in detail how the individual tasks in the stack of active tasks T determine what actions, by what agents, can coherently occur next in a dialogue with COREF. In Section 3.3.4.1.1, we discuss the model of agent actions used in COREF. In Section 3.3.4.1.2, we present the task models COREF uses to organize sequences of these actions into the various subtasks that COREF can take up. Finally, in Section 3.3.4.1.3, we show how COREF exploits these models to frame its object identification game as a two-agent task-oriented activity.

3.3.4.1.1 COREF’s model of agent actions The various tasks that COREF can perform are defined in terms of a set \mathcal{A} of possible *action types*. Each action type $\alpha \in \mathcal{A}$ is formalized parametrically as $\alpha = \mathbf{f}[p_1, \dots, p_n]$, where \mathbf{f} is an identifying symbol for the action type, and p_1, \dots, p_n are free parameter variables. (We will use square brackets in action terms, to make them easier to distinguish from propositional terms or mathematical functions.)

Each action type is marked as either *tacit* or *public*. (See Section 2.5.1.) This distinction is made as follows. Each action type can optionally be linked to a motor action, which we view as *generating* that action (Goldman, 1970; Pollack, 1986). For example, dialogue acts are linked to a **say**[s] motor action, in which utterance s is performed. Generally, purely mental actions are not linked to any motor action. If an action type α is not linked to any motor action, or if the motor action to which an action type is linked is *tacit*, then α is marked as *tacit*. Otherwise, α is *public*.

An actual action a by an agent x takes the form $a = \sigma(\alpha)$, where $\alpha \in \mathcal{A}$ and σ is a variable assignment that instantiates the free parameters of α . An instantiated action a , when performed by x in context c , effects a *deterministic* transformation on the current dialogue context. We formalize this by way of an update function:

$$c' = \text{UPDATE-A}(x, a, c) \tag{3.3}$$

The UPDATE-A function is part of COREF’s Update control module, and is described further in Section 3.3.8.1 (page 119).

The COREF action set \mathcal{A} comprises about 20 action types. We now give a few examples.

There is an action type, **pushCollabRef**[D, M, T], in which a director D initiates collaborative reference, with matcher M , to a target T . This action type is *tacit*: we identify its occurrence with the mental decision by D to collaboratively identify target T to M . The update associated with this action type is to push a new task onto the stack of active tasks in the context.

Once a collaborative reference task for some target T is underway, in order to help the matcher

identify T , the director can perform a dialogue act, `addcr`[T, C], which performs the update of adding an additional constraint C to the constraint network associated with T .

The identification by the matcher that the target T is some visual object R is captured by a tacit mental action `setVarValue`[T, R]. As a purely mental action, `setVarValue` is not linked to any motor action. Its effect is to add the proposition `varValue`(T, R) to the context. After identifying R as the correct object, the matcher can take the tacit action `addToScene`[R]. This action is linked to the (tacit) motor action `clickToAdd`[R]. Intuitively, by taking the motor action of clicking the object, the matcher thereby adds it to the scene. The effect of `clickToAdd`[R] is to physically move the object into the scene part of the matcher’s experiment interface. The update associated with agent A taking action `addToScene`[R] is that the proposition `inScene`[R, A] is added to the list of propositions in the context.

3.3.4.1.2 COREF’s models of dialogue subtasks COREF represents the individual tasks it can participate in using a custom state machine model which we call a *graph task*. A graph task is conceptually similar to a finite state machine: it consists of a finite set of states and a collection of labeled transitions between states. The primary difference between a graph task and a finite state machine is that a graph task provides additional machinery to allow a logical vocabulary of constants, variables, and complex terms to parameterize both states and state transitions.

We will illustrate our graph task model using the model `CollabRef`(D, M, T) in Figure 3.7.

In a graph task, transitions labels can take several forms. An action transition label takes the form `Agent does Action`. `Agent` is an arbitrary logical term – typically either a constant designating a specific agent (e.g., `a9` or `COREF`), or a role variable designating the agent playing a specific role. `Action` is a logical term that parameterizes an action. An action transition can occur only when an agent consistent with the `Agent` term takes an action consistent with the `Action` term.

For example, in the `CollabRef`(D, M, T) task in Figure 3.7, there is a transition between state `start2` and state `end_events1` labeled `D does addcr`[T, P]. This transition permits the director D to coherently perform the dialogue act `addcr`[T, P] when the current state of the graph task is `start2`.

A second kind of transition label places a propositional constraint on the global dialogue state within which the graph task is active. We notate such a constraint `[c] q`, and define it to be true iff the context $c = \langle R, P, T, C, U, O, H \rangle$ and $q \in P$. For example, in the `CollabRef`(D, M, T) task, there is a transition between states `start_until1` and `end_until1` labeled `[c] varValue`(T, R). This transition can occur only if there is a proposition of the form `varValue`(T, R) in the context.

graph task definition:

```

graphTask CollabRef(D,M,T) {
  until (varValue(T,R)) {
    D does addcr[T,P]
    or M does setVarValue[T,R]
  }
}

```

compiled graph task:

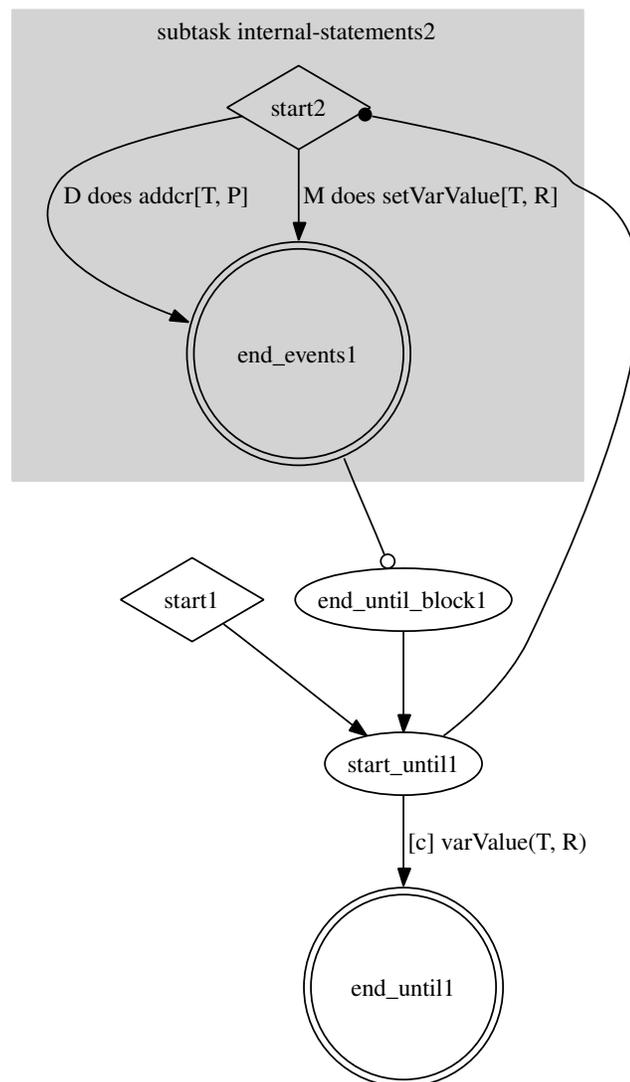


Figure 3.7: COREF's CollabRef (D,M,T) task. In this task, the director D helps the matcher M identify a target T as visual object R.

In the ordinary evolution of a `CollabRef(D,M,T)` task, the matcher eventually identifies the target T as visual object R , i.e. the matcher eventually performs the tacit action `setVarValue(T,R)`. One of the updates caused by this tacit action is to add the proposition `varValue(T,R)` to the context.

Finally, empty transitions are supported. These transitions can always occur. Note that the `CollabRef(D,M,T)` task contains several empty transitions.

Since transitions between states in a graph task can be associated with particular substitutions of values for variables, the current configuration of a graph task is characterized by a pair (s, σ) , where s is the current state of the graph task (`start2`, `end_events1`, etc.), and σ is the variable assignment under which state s was reached.⁸

COREF currently includes six possible graph tasks, which we present in Figures 3.7-3.13. These tasks are `CollabRef`, `CreateParallelScenes`, `ManageAmbiguity`, `WhichQ`, `YNQ`, and `Remind`. To facilitate the compact specification of these tasks, we have developed a custom formal language that supports the definition of graph tasks using loop constructs, action statements, propositional constraint statements, and nested statement blocks. A graph task definition expressed in this custom language is automatically compiled into the actual state machine model that COREF exploits at run time. Figures 3.7-3.13 present the original task definitions as well as the compiled versions of COREF’s graph tasks.

We will discuss these six graph tasks in the context of particular subdialogue examples in the remainder of the dissertation. For now, we have reached the point where we can state the form that the task stack component of COREF’s context representation takes: in any context $c = \langle R, P, T, C, U, O, H \rangle$, the task stack T takes the form $T = \langle (t_1, s_1, \sigma_1), \dots, (t_n, s_n, \sigma_n) \rangle$, where t_1, \dots, t_n are graph tasks and the internal state of graph task t_i is (s_i, σ_i) for $i = 1..n$.

3.3.4.1.3 COREF’s object identification game as a two-agent dialogue task We now define COREF’s object identification game as a two-agent dialogue task. In the model developed in Section 2.2, a two-agent task T takes the form $T = \langle S, I, R, A, N, U \rangle$. We let S , the set of possible task states, be the space of COREF dialogue states of the form $c = \langle R', P', T', C', U', O', H' \rangle$, as discussed in Section 3.3.4.1 (page 89).

⁸If all the possible variable assignments could be enumerated in advance, it would be possible to transform a graph task into a larger finite state model in which all the variable assignments have been “compiled out”. However, each new utterance that occurs in a dialogue with COREF introduces new possible variable assignments, so such a transformation is not practical. A second difference between graph tasks and finite state models lies in the dependence of propositional labels on the propositions that are present in the global dialogue state. It is not possible to compile out this dependence, because, in COREF, the propositions in the global dialogue state can change for reasons that are exogenous to any graph task. In effect, this makes the global dialogue state and the component graph tasks within it interdependent.

graph task definition:

```

graphTask CreateParallelScenes() {
  until (finishedCreateParallelScenes()) {
    currentTarget(T)
    director(D)
    matcher(M)
    escapeif (finished(T)) {
      D does pushCollabRef[D,M,T]
      varValue(T,R)
      M does addToScene[R]
      D does continueTask[T]
    }
  }
}

```

Figure 3.8: Definition of COREF’s `CreateParallelScenes()` task. In this task, two agents collaboratively refer to a sequence of target objects. After each collaborative reference subtask, the matcher adds the object to their scene, and the director clicks continue.

The initial state $I = \langle \emptyset, \emptyset, \langle \rangle, \emptyset, \emptyset, \emptyset, \langle \rangle \rangle$ is an “empty dialogue state” in which no objects are pending for identification, no propositions have been added to the context, no tasks are underway, no constraint networks are present, no objects or properties are present in the universe of discourse, no visual objects are present in the interface, and nothing has happened yet in the dialogue.

The set of roles $R = \{\text{COREF}, \text{User}\}$ is simply identified with the two participants in the object identification game. (The subtasks that are taken up over time may recast these two participants in different roles with respect to the subtasks. For example, COREF may play the role of *director* for one `CollabRef` subtask, and then play the role of *matcher* in a subsequent `CollabRef` subtask.)

The action set A is the space of instantiated actions $\sigma(\alpha)$ for $\alpha \in \mathcal{A}$.

The “coherent next actions” function $N : R \times S \rightarrow \mathcal{P}(A)$ is defined in terms of the stack of active tasks T in dialogue state S . If no tasks are underway (i.e. if $T = \langle \rangle$), then $N(R, S) = \emptyset$ for all R and S . Otherwise, let $T = \langle (t_1, s_1, \sigma_1), \dots, (t_n, s_n, \sigma_n) \rangle$, where (t_1, s_1, σ_1) describes the top graph task on the stack. (See Section 3.3.4.1.2 (page 92).) Then we have

$$N(R, S) = \{\sigma(a) \mid R \text{ does } a \text{ is an outgoing transition from } s_1 \wedge \sigma_1 \subseteq \sigma\} \cup \text{EXTRA-ACTIONS}(R, S)$$

where $\text{EXTRA-ACTIONS}(R, S)$ is a set of additional actions that R can perform in state S . These

compiled graph task:

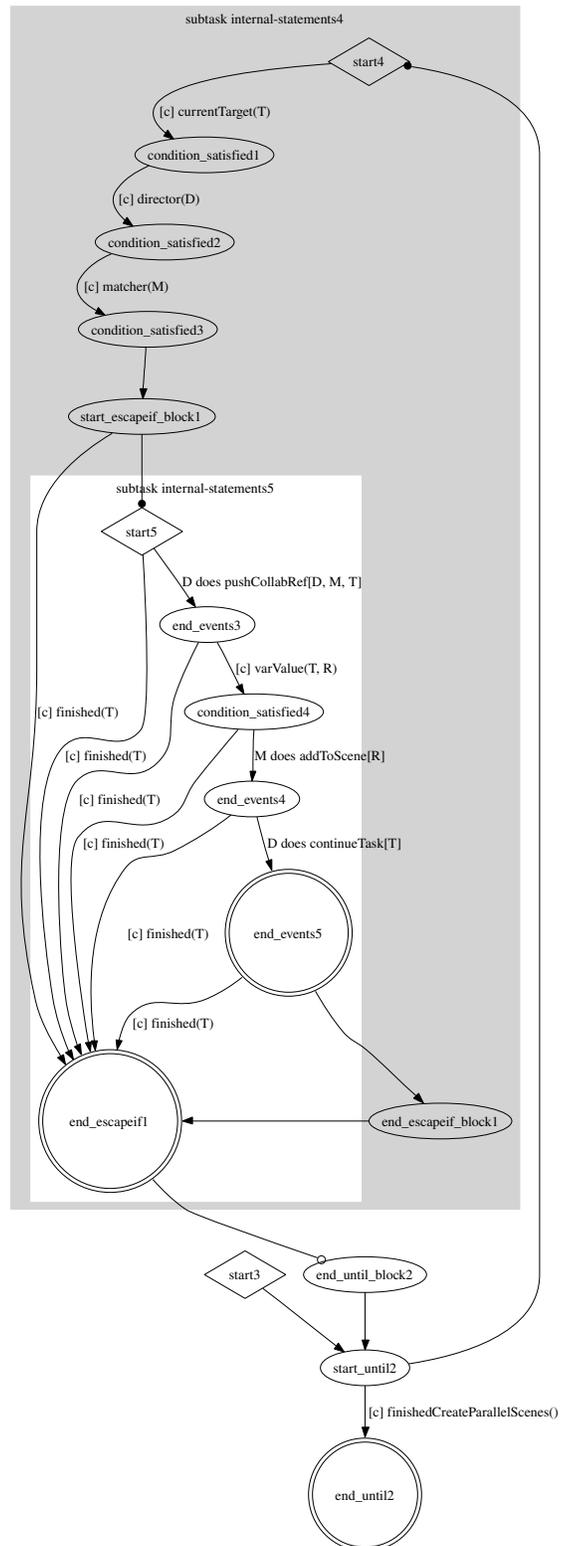


Figure 3.9: Compiled version of COREF's `CreateParallelScenes()` task.

graph task definition:

```

graphTask ManageAmbiguity(Observer,SE) {
    Observer does pushClarify[SE,Actor,Observer,T]
    or Observer does tacitNop[SE]
    or Observer does nop[SE]
    or Observer does flagProblematic[SE]
}

```

compiled graph task:

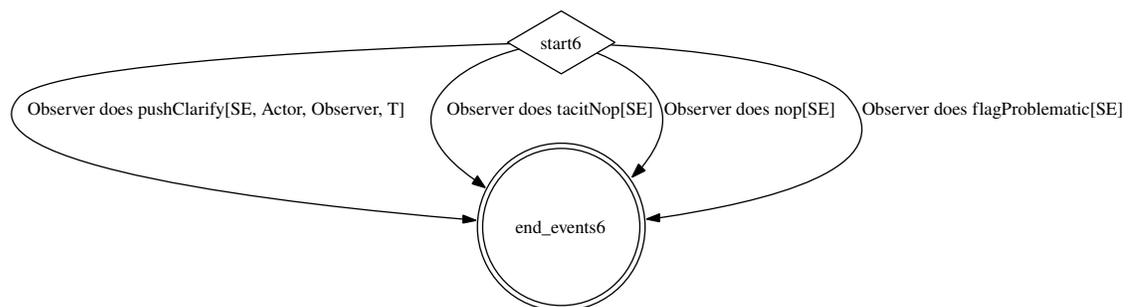


Figure 3.10: COREF's `ManageAmbiguity(Observer,SE)` task. In this task, `Observer` has the opportunity to take several actions in response to a sensory event `SE` in which `Observer` perceived an action by `Actor`.

graph task definition:

```
graphTask WhichQ(Asker,Knower,AnswerAction) {  
    Asker does askWhichQ[AnswerAction]  
    Knower does AnswerAction  
}
```

compiled graph task:

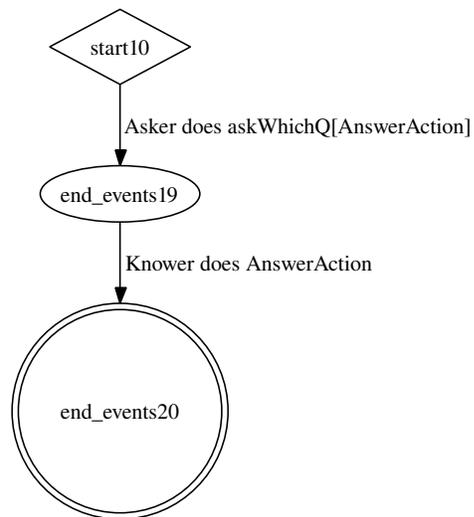


Figure 3.11: COREF's WhichQ(Asker,Knower,AnswerAction) task. In this task, **Asker** asks **Knower** a which question.

graph task definition:

```
graphTask YNQ(Asker,Knower,YesAction,NoAction) {  
    Asker does askYNQ[YesAction]  
    Knower does YesAction  
    or Knower does NoAction  
}
```

compiled graph task:

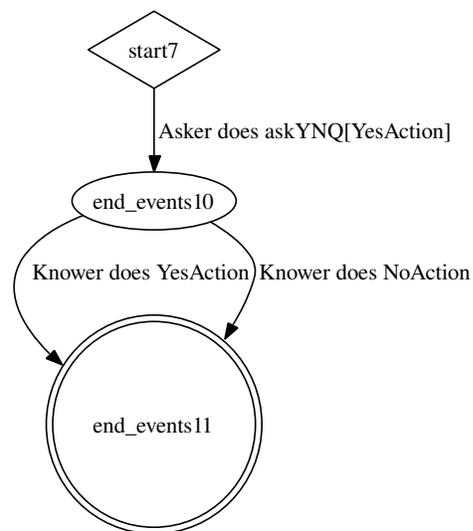


Figure 3.12: COREF's $YNQ(Asker,Knower,YesAction,NoAction)$ task. In this task, **Asker** asks **Knower** a yes/no question.

graph task definition:

```

graphTask Remind(Asker, Knower, Happened, NoAction, Action) {
  Asker does askYNQ[Happened[Knower, Action]]
  or Asker does command[Knower, Action]
  Knower does Happened[Knower, Action]
  or { Knower does Action
      Knower does Happened[Knower, Action]
    }
  or Knower does NoAction[Happened[Knower, Action]]
}

```

compiled graph task:

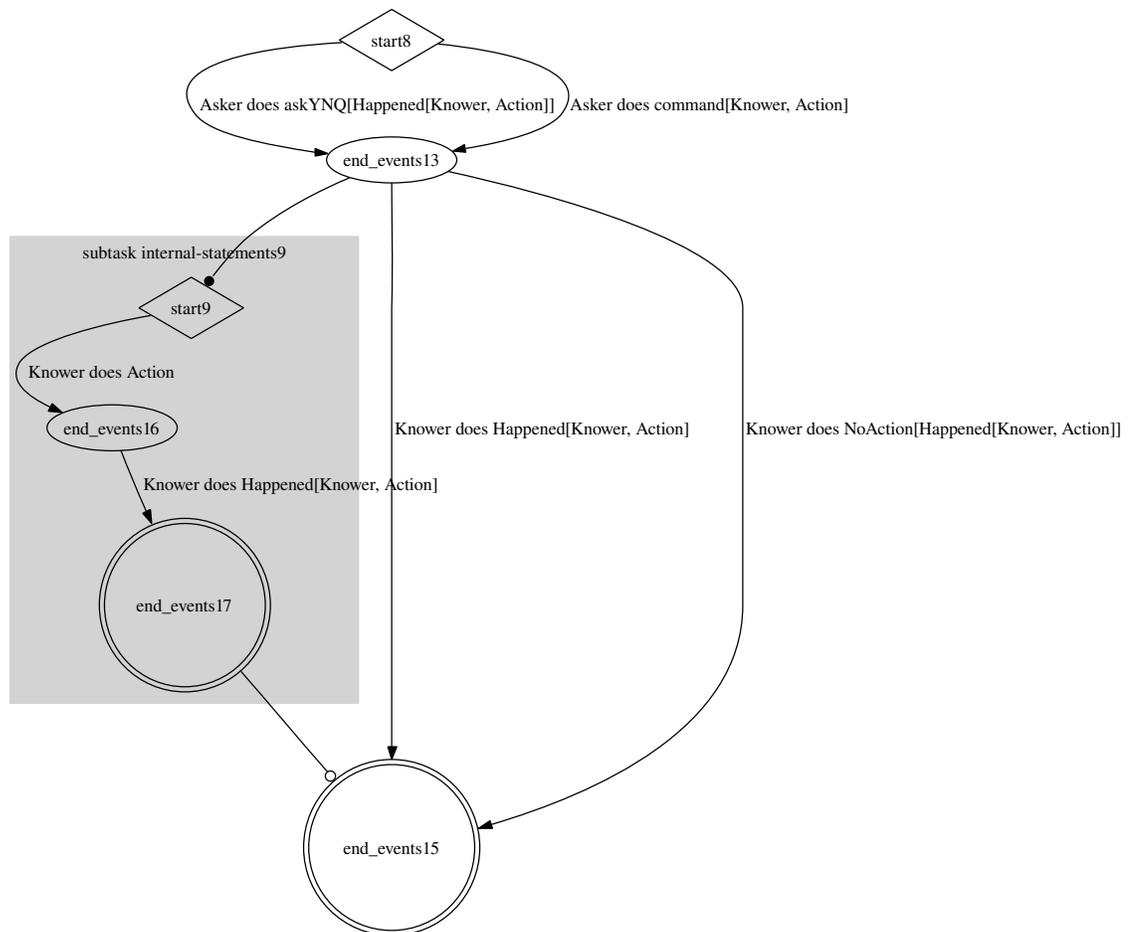


Figure 3.13: COREF's `Remind(Asker, Knower, Happened, NoAction, Action)` task. In this task, **Asker** asks **Knower** whether **Knower** has done an action, possibly as an indirect request or reminder to perform the action.

additional actions are determined by hand-built rules. COREF includes a small set of rules which allow the following actions to occur under certain conditions:

- `pushYNQ[Asker,Knower,YesAction,NoAction]` (a tacit task-pushing action),
- `pushWhichQ[Asker,Knower,AnswerAction]` (a tacit task-pushing action),
- `past[Agent,Action]` (a dialogue act),
- `skip[Target]` (a public action),
- `abandonTasks[N]` (a dialogue act that abandons, i.e. pops, the top-most N tasks), and
- `tacitAbandonTasks[N]` (a parallel tacit action).⁹

Generally, the EXTRA-ACTIONS mechanism allows COREF to support actions which seem to be able to happen coherently at a wide variety of points in a dialogue (for example, asking a yes/no question) without requiring that they be incorporated into complex graph task models.

Finally, the update function $U : R \times S \times A \rightarrow S$ for COREF's two-agent task is defined as follows:

$$U(R, S, A) = \begin{cases} \text{UPDATE-A}(R, A, S) & \text{if } A \text{ is tacit} \\ \text{UPDATE-MA}(x, o, \text{UPDATE-A}(R, A, S), P_x(I|o, s_t), i) & \text{if } A \text{ is public} \end{cases} \quad (3.4)$$

For tacit actions, the state is simply updated by the the UPDATE-A function (see Section 3.3.8.1 on page 119), which returns the dialogue state that reflects the updates directly associated with R performing action A in state S .

If A is a public action, then A triggers COREF's interpretation process, which causes additional updates to be made to the dialogue state. These additional updates allow COREF to keep track of its uncertainty in interpretation; they also push a `ManageAmbiguity` subtask, which makes it coherent for the observer x of action A to perform certain follow-up actions to manage their uncertainty. These updates are computed by the Update Control module's UPDATE-MA function (-MA for `ManageAmbiguity`); see Section 3.3.8.1 (page 119) for the details.

We can now see that COREF does indeed frame its object identification game as a two-agent task in the sense of Section 2.2. COREF exploits this framework in identifying a contributive action

⁹Because COREF associates public/tacit status with action *types*, in cases where the "same action" seems to appear in both tacit and public varieties, COREF includes two separate action types (one public and one tacit). The actions `abandonTasks[N]` and `tacitAbandonTasks[N]` provide one example of this modeling technique.

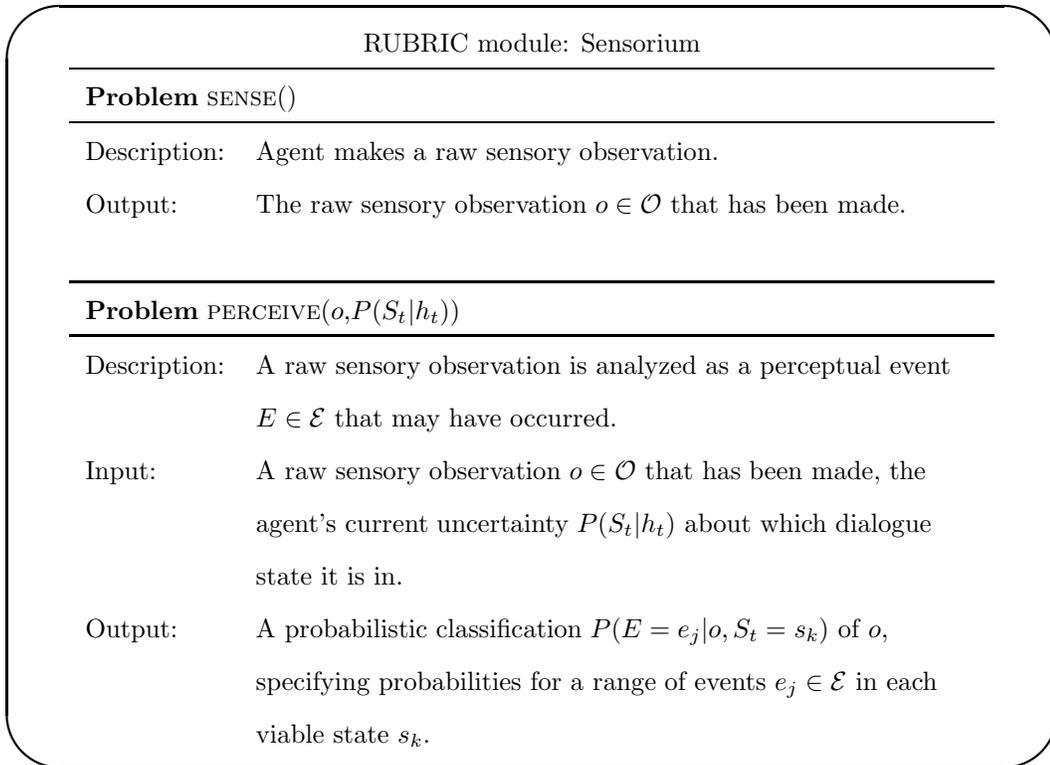


Figure 3.14: The Sensorium module.

to perform next in this two-agent task. We'll see how COREF identifies a contributive action to perform in Section 3.3.9.1 below.

For now, with these details of COREF's dialogue state representation under our belt, we continue our tour of the RUBRIC and COREF system architecture modules.

3.3.5 Sensorium

The RUBRIC Sensorium module is summarized in Figure 3.14. We assume there is some set \mathcal{O} of possible raw sensory observations the agent can make. For example, \mathcal{O} might be a space of possible audio samples from a microphone, or possible images from a camera. The agent's Sensorium is responsible for delivering these observations (SENSE) and for classifying them (PERCEIVE) into a set \mathcal{E} of perceptual events that are relevant to the agent's dialogue task. A perceptual event $E \in \mathcal{E}$ might be the word sequence identified by an automatic speech recognition algorithm, or a user motor action detected in a camera image, for example.

3.3.5.1 COREF's Sensorium

As discussed in Section 3.3.3.1 (page 87), there is no substantive perceptual uncertainty in the COREF agent. We therefore simply let $\mathcal{O} = \mathcal{E}$.¹⁰ COREF's Sensorium delivers the following observations:

$$\mathcal{O} = \mathcal{E} = \{ \text{ObservedAction}(\text{Agent}, \text{Action}), \\ \text{PerceivedCreateParallelScenesFinished}(), \\ \text{PerceivedFigureObjects}(\text{ObjectSet}), \\ \text{PerceivedNewReferenceProblem}(\text{SalientProperties}), \\ \text{PerceivedNewRoles}(\text{Director}, \text{Matcher}) \\ \}$$

COREF perceives an `ObservedAction(Agent, Action)` event whenever the user takes one of the following public motor actions: `ClickContinue()`, `ClickSkip()`, or `Say(S)`. (Note that COREF does not observe tacit motor actions performed by the user.)

The other perceptual events COREF experiences keep COREF up to date on the state of the game. For example, COREF experiences a `PerceivedCreateParallelScenesFinished()` event when there are no more objects remaining in its `CreateParallelScenes` task (see Figure 3.8 on page 95). This event is analogous to a pop-up dialogue box that informs the user when the game is finished.

COREF experiences a `PerceivedFigureObjects(ObjectSet)` event whenever the visual objects in its interface change. This happens, for example, when the continue button is pressed, or when COREF (as matcher) moves an object it believes to be the target object over into its scene. An example of this event occurs at time 7 in the example dialogue of Figure 3.3.

COREF experiences a `PerceivedNewReferenceProblem(SalientProperties)` event whenever the game advances to the next target object. The `SalientProperties` is a list of properties (colors, shapes, etc.) that are judged “salient” by a hand-built salience model in the new reference problem. An example of this event occurs at time 6 in the example dialogue of Figure 3.3.

Finally, COREF experiences a `PerceivedNewRoles(Director, Matcher)` event whenever the roles

¹⁰In COREF, as shown in Figure 3.5 (page 88), only the `SENSE` method is used. The `PERCEIVE` method is not used. Also, note that this means that the agent's uncertainty $P(S_t|h_t)$ plays no role in COREF's Sensorium.

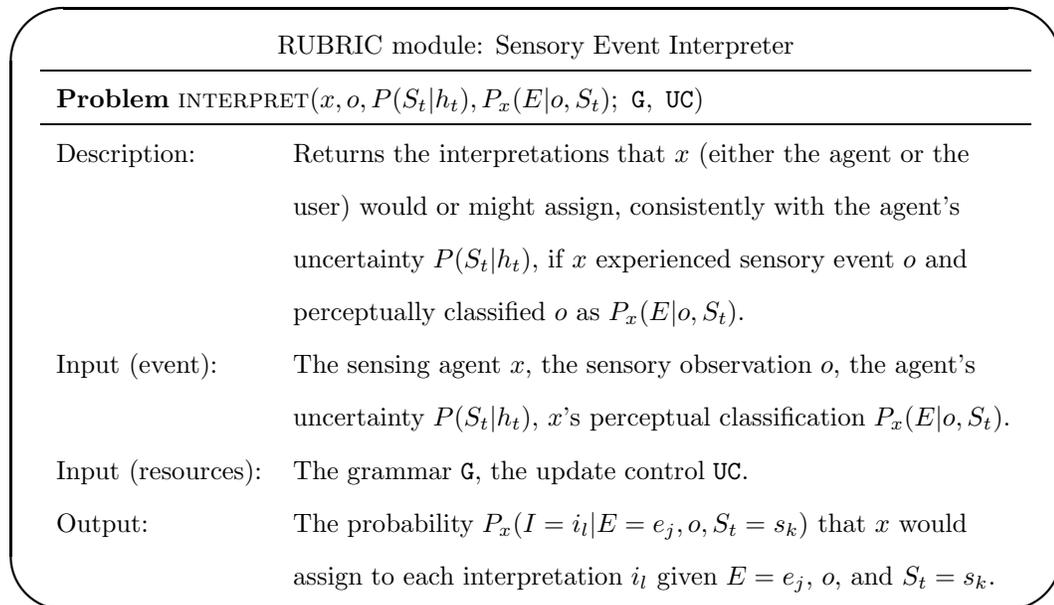


Figure 3.15: The Sensory Event Interpreter module.

in the game change; if `Director=COREF`, COREF will play the director. If `Matcher=COREF`, COREF will play the matcher. This event is analogous to a dialogue box that pops up to inform the user when the user's role changes.

3.3.6 Sensory Event Interpreter

The RUBRIC Sensory Event Interpreter module is summarized in Figure 3.15. The module is responsible for hypothesizing a set of interpretations for a given sensory event (e.g., an audio sample from a microphone) under alternative assumptions about the correct perceptual classification of the event (e.g., which words were uttered) and about the correct dialogue state (e.g., that the state is one in which the user is trying to identify a song they want the agent to play). In particular, this problem-solving identifies perceptual analyses and dialogue states that are consistent with the agent's information, constructs relevant interpretations for each set of assumptions, and assigns each interpretation a conditional probability. This process may yield zero, one, or more interpretations for a given sensory event.

The RUBRIC framework makes no assumptions about the exact form interpretations take. The rationale for requiring the existence of a module that delivers "interpretations", however, is that most dialogue systems exploit a level of analysis at which alternative *intentional* explanations for perceived

actions and utterances are entertained. In implementing contribution tracking, we identify these intentions with intended contributions. The Sensory Event Interpreter module therefore situates hypothesized contributions within a coherent probabilistic framework.

The main substantive requirement is that the Sensory Event Interpreter module be able to interpret sensory events “from either agent’s perspective.” In particular, not only can the agent use the module to interpret its own sensory events, for example by assigning intentional explanations to perceived user actions, but the agent can also use the module to interpret hypothetical sensory events that the *user* might experience. While it is of course difficult for one agent to represent another agent’s sensory experiences, and to fully capture their interpretive reasoning, this functionality is important in the RUBRIC framework because it makes the Sensory Event Interpreter a resource that can be exploited in generation, as the agent attempts to identify a contributive action to perform. Specifically, in assessing the acceptability of a possible agent action, the Intention Generator module can invoke the Sensory Event Interpreter in order to estimate the interpretations the user could be expected to assign to the agent’s action. To do so, we assume that the Sensory Event Interpreter can interpret a hypothetical user sensory event, given the user’s anticipated perceptual classification of the event. The output of the INTERPRET function, in this case, relates the anticipated user interpretations to the agent’s uncertainty about which dialogue state it is in.

Interpretation and generation are thus tightly coupled in a RUBRIC agent. This coupling creates a pressure for the interpretations that the agent assigns to user actions to take exactly the same form as the interpretations the agent assigns to its own actions; we will see that this is indeed the case in the COREF agent. This coupling also enforces a requirement that may help promote a kind of “reversibility” in the agent’s language competence, in that the agent must, by design, be able to interpret anything it can say itself. (While human interlocutors seem to have this competence, many state of the art dialogue systems do not.) When faced with human users that naturally “entrain” to the agent’s own vocabulary by using the same words the agent uses (see e.g. Pearson et al. (2006)), such an agent might be more likely to successfully interpret its human users’ utterances.

3.3.6.1 COREF’s Sensory Event Interpreter

In interpreting a sensory event o , COREF starts by making a binary decision about whether the event was caused by a public agent action or if it was instead caused by some other change in the game state (e.g., the appearance of new figure objects, a role change, etc.). Since COREF does not face any perceptual uncertainty, this binary decision is a given for COREF.

In interpreting a public action, COREF assigns interpretations in the form of action sequences

that it believes would constitute coherent contributions to the dialogue task.¹¹ Specifically, an interpretation i takes the form of an action sequence $i = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$ in which agent A_1 performs action a_1 , then agent A_2 performs action a_2 , and so on. We will use the words *interpretation*, *intention*, and *contribution* interchangeably in discussing these sequences.

Before discussing how COREF identifies possible interpretations, we note two broad features that all its interpretations share. First, the sequence always consists of a prefix of zero or more tacit actions, followed by one or more public actions or dialogue acts. When a dialogue act is preceded by tacit actions in an interpretation, we view the utterance as implicating those tacit actions. Similarly, when a non-verbal, public motor action is preceded by tacit actions in an interpretation, we view the motor action as implicating those tacit actions. We will see that these implicatures are an important part of the interlocutors' coordination in COREF's dialogues. Second, both agents can be assigned actions within a single interpretation: in an interpretation $i = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$, sometimes $A_j \neq A_k$ for $j \neq k$. In these cases, we view the agent whose public action is being interpreted as signaling or implicating that the other agent has taken one or more tacit actions. We will present examples that illustrate both these features shortly.

At a high level, COREF treats interpretation as a problem of abductive intention recognition (Thomason et al., 2006; Hobbs et al., 1993). Here, we extend the characterization of abductive intention recognition of Thomason et al. (2006) to support uncertainty about the current context.

COREF abductively recognizes the intention i of an actor A in three steps. First, for each context s_k that was viable at the time the public action was performed, COREF builds a representation which we will call a *horizon graph*. A horizon graph uses the pending tasks in a specific context to circumscribe the set of alternative intentions that A might try to contribute from that context. In the second step, COREF uses the horizon graph and other resources to solve any constraints associated with the observed action. This step instantiates any free parameters associated with the action to contextually relevant values. For utterances, the relevant constraints are identified using a grammar that determines both presupposed constraints — which must hold in the context — as well as schematic dialogue acts — which must be coherent in the context. Each successful instantiation of the action's parameters determines a possible interpretation. In the third step, COREF uses a probabilistic model to assign a probability $P(I = i | o, S_t = s_k)$ for each interpretation i and assumed context s_k .

We now present further details of this three step interpretation process using the subdialogue

¹¹Recall that the use of action *sequences* rather than single actions was motivated and incorporated into our definitions of contributive action and language use in Chapter 2.

ManageAmbiguity(s1, [COREF does say[the blue circle]])
CollabRef(COREF, s1, t3)
CreateParallelScenes()

Figure 3.16: The task stack in state s2106.

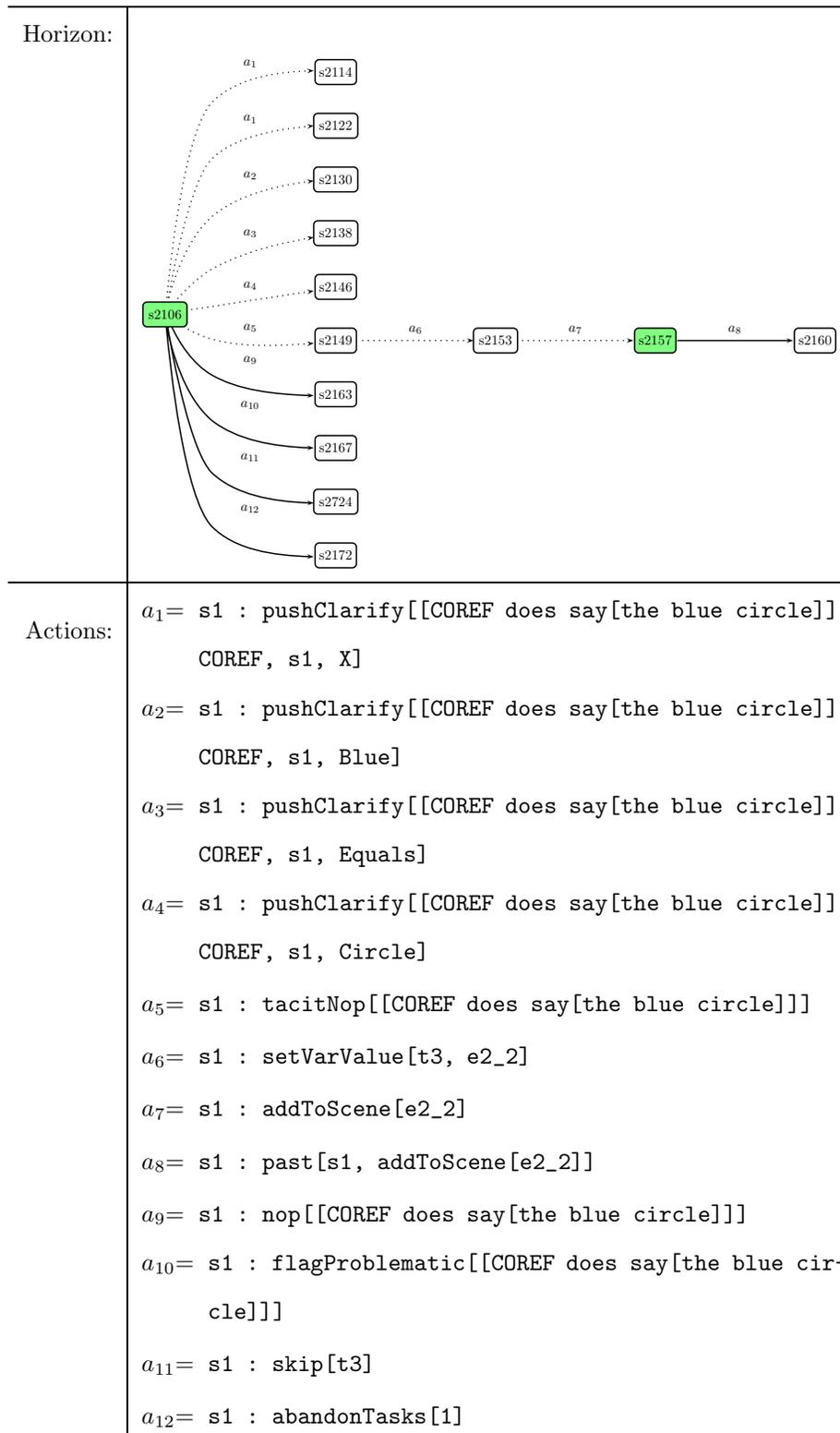
in Figure 3.3 on page 3.3. In this subdialogue, COREF initiates collaborative reference to a new target by saying *the blue circle* at time 1. COREF’s human interlocutor, s1, responds by saying *ok* at time 2. At time 2, before the user said *ok*, COREF was certain about the dialogue state, and in particular assigned $P(S_2 = s2106) = 1$. In state s2106, three tasks are underway, organized on a stack as illustrated in Figure 3.16.¹² The topmost task on the stack is a `ManageAmbiguity` task in which s1 is managing his ambiguity in interpreting COREF’s utterance *the blue circle*. Lower on the stack, COREF is directing s1 in a `CollabRef` task for target object `t3`. Lower still, the two are engaged in a `CreateParallelScenes` task that corresponds to their overall multi-object identification game.

3.3.6.1.1 Building the horizon graph In interpreting an utterance, like *ok* here, COREF can exploit its task models to determine all the various actions that could coherently come next. Some of these actions are tacit actions, and so in general COREF can determine all the various interpretations which terminate in a public action that might be consistent with its sensory observation $o = s1 \text{ does say}[ok]$. COREF represents these sequences in the horizon graph in Figure 3.17.

Figure 3.17: Horizon graph for s1: *ok*

continues on next page...

¹²See Figure 3.6 (page 90) for more details about state s2106.

Figure 3.17: Horizon graph for s1: *ok* (continued)

In the figure, nodes are dialogue states, and edges are action descriptions of the form **Agent** : **ac-**

tion. Dashed edges correspond to tacit actions. For example, starting from state s2106, one coherent action would be the dialogue act $a_9 = s1 : \text{nop}[[\text{COREF does say}[\text{the blue circle}]]]$. This “no operation” action is the way COREF handles acknowledgments like *ok*, *okay*, *alright*, etc. We use the “no operation” vocabulary because the `nop` action has no effect on the agent’s dialogue state, other than that its performance is one way to complete the `ManageAmbiguity` task, an example of which is on the top of the stack in state s2106.¹³ Conceptually, we view a speaker who acknowledges an utterance by saying *ok* as merely declining an opportunity to manage any uncertainty they may face by initiating a follow-up dialogue, rather than marking the interpretation of the acknowledged utterance as common ground (Traum, 1994).

It is well known that an utterance of *ok* can mean many different things, even in specific dialogue contexts (Core and Allen, 1997). In COREF’s domain, for example, the matcher can (and often does) say *ok* to convey both that they have identified the target object and that they have added it to their private scene. This interpretation shows up in this horizon graph as a sequence of actions:

```

⟨a5 = s1 : tacitNop[[COREF does say[the blue circle]]],
a6 = s1 : setVarValue[t3, e2_2],
a7 = s1 : addToScene[e2_2],
a8 = s1 : past[s1, addToScene[e2_2]]⟩

```

In this sequence, s1 first performs a `tacitNop` of COREF’s previous utterance (i.e. tacitly declines the opportunity to manage their uncertainty about *the blue circle*), then identifies the target as object `e2_2` (`setVarValue`), then adds `e2_2` to their private scene (`addToScene`), and then finally asserts that they have added `e2_2` to their private scene (by performing the dialogue act `past[s1, addToScene[e2_2]]`). Using its task models, COREF judges this entire action sequence to be coherent starting from s2106, and so includes it as part of its horizon graph.

For efficiency reasons, not all “coherent” action sequences make it into COREF’s horizon graph for any given sensory event. For example, another coherent tacit action from state s2106 is $a_1 = s1 : \text{pushClarify}[[\text{COREF does say}[\text{the blue circle}], \text{COREF}, \text{s1}, \text{X}]$. This action pushes a clarification subtask onto the stack – specifically, one in which s1 seeks to clarify the referent `X` of COREF’s expression *the blue circle*.¹⁴ This tacit action is coherent due to the presence of the `ManageAmbiguity` task on the top of the stack in state s2106. However, it seems clear that s1 is not starting a clarification task when he says *ok*.

¹³The details of the `ManageAmbiguity` task are shown in Figure 3.10 (page 97).

¹⁴We actually model such a clarification subtask using our `CollabRef` task. Specifically, the `pushClarify` action pushes a `CollabRef` task in which the original speaker collaboratively identifies their original meaning to the original addressee.

Generally, COREF needs a strategy for constructing the horizon graph efficiently, because there are many tacit actions that *could* be performed coherently, according to the agent’s task models, but which are irrelevant to the particular utterance at hand. COREF currently solves this problem using hand-built rules that allow it to prune out specific parts of the horizon graph. These rules allow COREF to abort further consideration of a_1 , and various other `pushClarify` actions visible in the figure, as irrelevant to the interpretation of an utterance of *ok*.¹⁵ COREF employs additional rules that limit the total depth of the horizon graph.

The states shaded in green in the figure constitute the *horizon* (Thomason et al., 2006) from state `s2106`. The horizon $Z(s_k)$ for a given state s_k is the set of states that are (i) reachable from s_k by a sequence of zero or more coherent tacit actions and (ii) from which a public action is coherent. Conceptually, the horizon is the set of states from which an observed action or dialogue act should be a coherent next action. The horizon therefore plays a special role in the second step of interpretation, in which constraints associated with the coherent performance of an action are checked. In particular, if the actor has performed a public action p at time t , an attempt is made to solve the constraints associated with p at each state $s \in Z(s_k)$ such that $P(S_t = s_k) > 0$.

3.3.6.1.2 Solving an action’s constraints If p is a natural language utterance, COREF consults its grammar in order to determine the constraints associated with the utterance. COREF uses a hand-built, lexicalized tree-adjoining grammar. The specific grammar formalism is TAGLET (Stone, 2002), a variant of tree-adjoining grammar in which the full adjunction operation is replaced with left and right sister adjunction, and in which lexical entries are linked to semantic constraints on variable values. COREF’s grammar allows two kinds of constraints to be associated with each lexical entry. Presuppositions are constraints which must already be true in the (horizon) state in which the utterance is evaluated. Dialogue act constraints are schematic dialogue act terms that must unify with a dialogue act that is coherent in the (horizon) state in which the utterance is evaluated. The overall constraints associated with an utterance are determined by performing a bottom-up chart parse of the utterance, and joining the presuppositions and dialogue acts associated with each edge in the chart.

For example, Figure 3.18 shows one of COREF’s two lexical entries for the word *ok*. In this entry, *ok* is assigned a simple syntactic analysis that treats it as a declarative sentential constituent which is not referential. The entry specifies a single presupposition, `ok-f(M,0)`, which requires that each

¹⁵They do so on the basis that *ok* cannot be grammatically analyzed as a question. Because the ultimate dialogue act performed by *ok* cannot, therefore, be one of the questioning dialogue acts, the rules are able to prune out all of the questioning dialogue acts – and these are the only acts that can coherently follow a `pushClarify`.

Syntax:	$\left[\begin{array}{l} \text{cat: } \mathbf{s}, \quad \text{stype: } \mathbf{dec}, \\ \text{reftype: } \mathbf{none} \end{array} \right]$ <p style="text-align: center;"> </p> <p style="text-align: center;">ok</p>
Operation:	comp [cat: s, stype:dec,reftype:none]
Presuppositions:	ok-f($M,0$)
Dialogue acts:	$M[]$

Figure 3.18: A lexical entry for *ok*

use of *ok* occur in a context that provides some dialogue act M , of grammatical arity 0, that can be performed by uttering *ok*. The entry also specifies that a single dialogue act, $M[]$, is performed by an utterance of *ok*.

To solve the constraints associated with an utterance, COREF first solves the utterance’s presuppositions, then tries to extend these solutions in a way that also solves the utterance’s dialogue acts. This process yields zero or more variable assignments under which both the presuppositions and dialogue acts are solved. We now discuss this process in some detail.

Presupposed constraints are solved using a hand-built domain model which directly associates particular constraints, like $\text{ok-f}(M,0)$, with algorithms that deliver assignments to variables (like M). In the case of the presupposition $\text{ok-f}(M,0)$, the domain model delivers a single solution: $M \leftarrow \text{nop}[O]$. Here, $\text{nop}[O]$ is what we call an *arity-bridging* function term. COREF uses arity-bridging function terms for several specific kinds of linguistic constructions. Such function terms allow COREF to interpret constraints even when the number of grammatical arguments to a predicate or function differs from the number of arguments present in the corresponding application predicate or function.¹⁶

In this particular case, *ok* corresponds to a dialogue act, nop , which we have modeled (in our `ManageAmbiguity` task model) as taking a single argument, viz., the sensory event whose uncertain interpretation is being managed. However, in English, one typically just says *ok*, with no additional grammatical argument — rather than saying something like *ok to your last utterance*, in which reference is made to the material being acknowledged.¹⁷ By solving the presupposition $\text{ok-f}(M,0)$ with the assignment $\sigma = \{M \leftarrow \text{nop}[O]\}$, a variable O is introduced which can be further instantiated when the utterance’s dialogue act is reconciled with COREF’s task models.

¹⁶Generally, our experience is that the cause of such discrepancies is that some entity that is important in the domain or task models is not expressed linguistically.

¹⁷Another way to see this issue is to compare *ok* to *roger that*.

Syntax:	$\left[\begin{array}{l} \text{cat: } \mathbf{s}, \quad \text{stype: } \mathbf{dec}, \\ \text{reftype: } \mathbf{none} \end{array} \right]$ <p style="text-align: center;"> </p> <p style="text-align: center;"><i>ok</i></p>
Operation:	comp [cat: s, stype:dec,reftype:none]
Presuppositions:	simplepast-f(Past,2), speaker-p(Speaker), Speaker(S)
Dialogue acts:	Past[S,TA]

Figure 3.19: A second lexical entry for *ok*

Under σ , the utterance’s dialogue act, $M[]$, is $\sigma(M[]) = \text{nop}[O] []$. To solve this dialogue act, because it contains an arity-bridging function term, COREF first rewrites the dialogue act simply as $\text{nop}[O]$, eliminating the “arity bridge”.¹⁸ It then attempts to unify $\text{nop}[O]$ with dialogue acts appearing in the horizon graph. In particular, COREF attempts to unify this term with each dialogue act that follows each horizon state $s \in Z(s2106)$. This unification succeeds exactly once, under assignment $O \leftarrow [\text{COREF does say}[\text{the blue circle}]]$, due to the coherence of action $a_9 = s1 : \text{nop}[\text{COREF does say}[\text{the blue circle}]]$ from horizon state $s = s2106$. This leads COREF to hypothesize, as one of its interpretations for $s1$ ’s utterance of *ok*, the intention:

$$i_{2,2} = \langle s1 : \text{nop}[[\text{COREF does say}[\text{the blue circle}]]] \rangle$$

COREF also finds a second intentional explanation of $s1$ ’s utterance:

$$i_{2,1} = \langle s1 : \text{tacitNop}[[\text{COREF does say}[\text{the blue circle}]]], \\ s1 : \text{setVarValue}[\text{t3}, \text{e2_2}], \\ s1 : \text{addToScene}[\text{e2_2}], \\ s1 : \text{past}[s1, \text{addToScene}[\text{e2_2}]] \rangle$$

This interpretation is derived using COREF’s second lexical entry for *ok*, which is given in Figure 3.19. This entry differs from the first in that it allows a speaker to assert that they have taken some action in the past. COREF’s chart parse of *ok* delivers two analyses, one using each of these entries. The constraints associated with the second entry are satisfied at state $s2157 \in Z(s2106)$, leading COREF to hypothesize $i_{2,1}$ as an alternative explanation for $s1$ ’s utterance.

COREF thus faces uncertainty, in interpreting $s1$ ’s utterance of *ok*, about which of two contributions $s1$ has made: $i_{2,1}$ or $i_{2,2}$. In the remainder of the dialogue in Figure 3.3, COREF tracks these two contributions.

¹⁸Generally, this rewriting process involves combining grammatically expressed arguments with variables for grammatically unexpressed arguments to arrive at a list of arguments isomorphic to the domain model’s argument structure.

Syntax:	$\left[\begin{array}{l} \text{cat: } \mathbf{s}, \quad \text{stype: } \mathbf{dec}, \\ \text{reftype: } \mathbf{none} \end{array} \right]$ <p style="text-align: center;"> </p> <p style="text-align: center;">yes</p>
Operation:	comp [cat: s, stype: dec, reftype: none]
Presuppositions:	speaker-p(Speaker), Speaker(S), ymq-p(YNQ,3), YNQ(S,Yes,No)
Dialogue acts:	Yes

Figure 3.20: COREF’s lexical entry for *yes*

We will give one more example of how COREF uses its grammar to construct interpretations for utterances. After s1’s utterance of *ok*, COREF responds to its perceived ambiguity by asking *did you add it?* (We will discuss how COREF chooses this utterance in Section 3.3.9.1, below.) In response to this question, s1 says *yes* at time 4. COREF’s lexical entry for *yes*, another one-word utterance, is depicted in Figure 3.20. At time 4, due to the previous perceived ambiguity about *ok*, COREF is uncertain whether the dialogue state is s2479 or s2465. COREF therefore attempts to interpret *yes* in each of these states. Figures 3.21 and 3.22 show COREF’s horizon graphs for these two states, respectively. From state s2479 (Figure 3.21), in which s1 previously meant $i_{2,2}$ by *ok* (the simple *nop* interpretation), COREF is able to assign interpretation $i_{4,2}$ to *yes*:

$$i_{4,2} = \langle s1 : \text{tacitNop}[[\text{COREF does say}[\text{did you add it ?}]]], \\ s1 : \text{addToScene}[\text{e2_2}], \\ s1 : \text{past}[s1, \text{addToScene}[\text{e2_2}]] \rangle$$

According to interpretation $i_{4,2}$, by saying *yes*, s1 has contributed that they have decided not to address any ambiguity in COREF’s question, added object *e2_2* to their scene, and asserted that they have done so. In particular, on this interpretation, s1 has just now added the object. Note that the horizon graph requires s1 to perform the tacit action *addToScene*[*e2_2*] before asserting that that action occurred in the past; this reflects COREF’s assumption that its human interlocutors are always truthful.

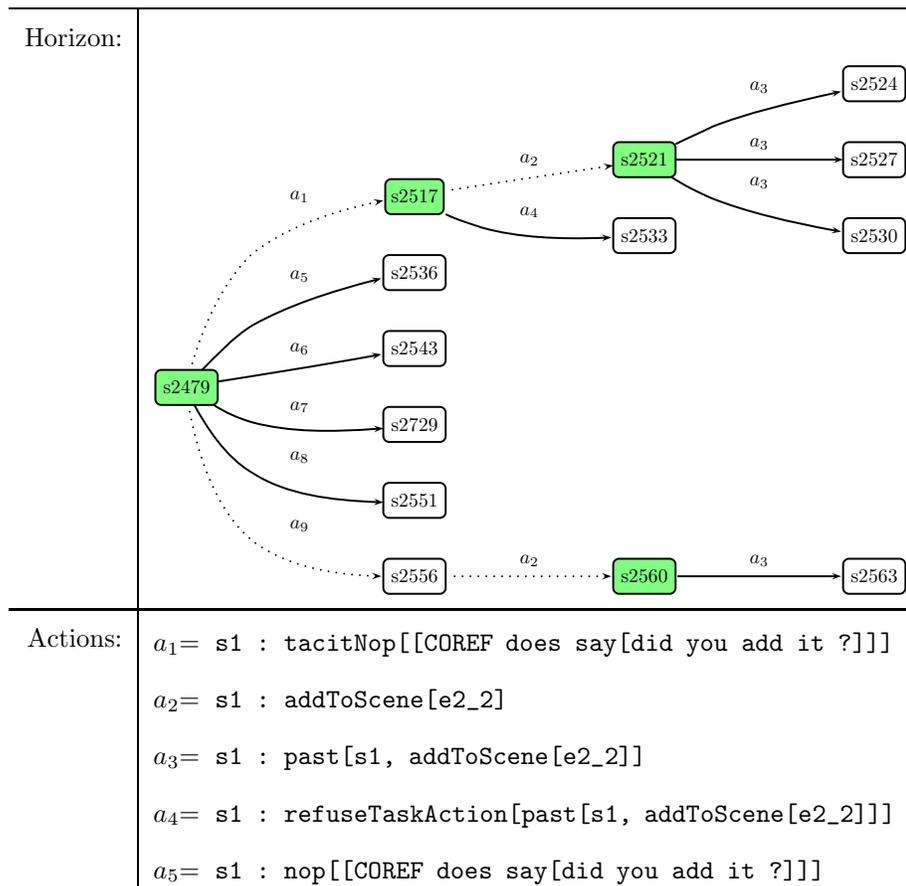
In state s2465 (Figure 3.22), on the other hand, s1 previously meant $i_{2,1}$ by *ok*. Under that interpretation, s1 had already contributed that they had added the object to their scene. Consequently, the horizon graph from state s2465 is different, and COREF is able to assign the following interpretation, $i_{4,1}$, to *yes*:

$$i_{4,1} = \langle s1 : \text{tacitNop}[[\text{COREF does say}[\text{did you add it ?}]]], \\ s1 : \text{past}[s1, \text{addToScene}[e2_2]] \rangle$$

According to interpretation $i_{4,1}$, by saying *yes*, s1 has contributed that they have decided not to address any ambiguity in COREF's question, and asserted that they have added the object to their scene. These two interpretations, therefore, reflect alternative coherent series of contributions that s1 may have made in their previous two utterances. COREF's contribution tracking allows it to track these contributions across the subdialogue. We will see in our discussion of COREF's Intention Generator module in Section 3.3.9.1 that, as a result of this uncertainty, COREF is a little uncertain about what contributions its *own* utterances are making, and employs the same kind of reasoning to track its own contributions as it does to track its user's.

We will give additional examples of the interpretations hypothesized by COREF's Sensory Event Interpreter in Chapter 4.

Figure 3.21: Horizon graph for state s2479.



continues on next page...

Figure 3.21: Horizon graph for state s2479. (continued)

$a_6 = s1 : \text{flagProblematic}[[\text{COREF does say}[\text{did you add it ?}]]]$ $a_7 = s1 : \text{skip}[\text{t3}]$ $a_8 = s1 : \text{abandonTasks}[2]$ $a_9 = s1 : \text{tacitAbandonTasks}[2]$

Figure 3.22: Horizon graph for state s2465.

Horizon:	<pre> graph LR s2465[s2465] -.- a1 --> s2482[s2482] s2465 -- a2 --> s2485[s2485] s2465 -- a3 --> s2488[s2488] s2465 -- a4 --> s2495[s2495] s2465 -- a5 --> s2728[s2728] s2465 -- a6 --> s2503[s2503] s2465 -.- a7 --> s2508[s2508] </pre>
Actions:	$a_1 = s1 : \text{tacitNop}[[\text{COREF does say}[\text{did you add it ?}]]]$ $a_2 = s1 : \text{past}[s1, \text{addToScene}[\text{e2_2}]]$ $a_3 = s1 : \text{nop}[[\text{COREF does say}[\text{did you add it ?}]]]$ $a_4 = s1 : \text{flagProblematic}[[\text{COREF does say}[\text{did you add it ?}]]]$ $a_5 = s1 : \text{skip}[\text{t3}]$ $a_6 = s1 : \text{abandonTasks}[2]$ $a_7 = s1 : \text{tacitAbandonTasks}[2]$

3.3.6.1.3 Assigning probabilities to interpretations Once COREF has identified a set of interpretations $\{i_{t,1}, \dots, i_{t,n}\}$ for a sensory event o at time t , the third step its Sensory Event Interpreter performs is to assign a probability $P(I = i_{t,j}|o, S_t = s_k)$ for each interpretation. In general, we conceive of this assignment as a process of weighted abduction (Hobbs et al., 1993; Thomason et al., 2006): there are competing interpretations, and the agent needs to weigh the different assumptions that went into constructing each interpretation in order to assess the likelihood that each interpretation is the correct one.

We have used two different probability models to date. The first is a hand-built model which simply assigns probability inversely proportional to the “size” of the intention:

$$P(I = i_{t,j}|o, S_t = s_k) \propto \frac{1}{\text{NUM-TACIT-ACTIONS}(i_{t,j}) + 1}$$

This hand-built model measures the size of an intention as the number of tacit actions in the intention, plus 1 for the single public action that concludes each intention.¹⁹ This hand-built model was initially used, in the absence of detailed empirical data, as a simple numeric way to prefer “simpler” interpretations. For example, this model views $i_{2,2}$ (which has 0 tacit actions) as more likely to be the correct explanation of s_1 ’s utterance of ok than $i_{2,1}$ (which has 3 tacit actions). The hand-built model’s probabilities have been used to color the state nodes in Figure 3.3.

The second model is based on numerous weighted features of intentions and the dialogue states in which they are hypothesized; the weights were learned from data harvested from COREF’s interactions with its users. The learned model is described in Chapter 4, Section 4.4.2.

We now continue our tour of the RUBRIC and COREF system architectures.

3.3.7 Grammar

The RUBRIC architecture assumes that there is some grammar or language model that is shared between the Sensory Event Interpreter and Intention Generator modules. However, RUBRIC does not stipulate the form that the grammar or language model takes.

3.3.7.1 COREF’s Grammar

COREF’s grammar formalism was discussed in Section 3.3.6.1.2 (page 110). In total, COREF’s grammar contains 379 hand-built lexical entries of the kind depicted in Figures 3.18, 3.19, and 3.20.

¹⁹Note that when the public action is an utterance, this measure always counts the utterance as contributing 1 to the size of the intention, even if the intention analyzes the utterance as performing more than one dialogue act.

3.3.8 Update Control

The RUBRIC Update Control module is summarized in Figure 3.23. The Update Control module’s FILTER problem is to compute $P(S_{t+1}|h_{t+1}) = P(S_{t+1}|o, h_t)$ as a function of the agent’s prior uncertainty and the given perception and interpretation of o by agent x . In this section, we define this computation.

By definition, we have:

$$P(S_{t+1}|h_{t+1}) = P(S_{t+1}|o, h_t) = \frac{P(S_{t+1}, o, h_t)}{P(o, h_t)} \propto P(S_{t+1}, o, h_t) \quad (3.5)$$

Assuming exactly one state $s_t \in v_t$ is the true dialogue state, we have:

$$\begin{aligned} P(S_{t+1}, o, h_t) &= \sum_{s_t \in v_t} P(S_{t+1}, o, h_t, s_t) \\ &= \sum_{s_t \in v_t} P(h_t) P(s_t|h_t) P(o|h_t, s_t) P(S_{t+1}|o, h_t, s_t) \\ &= \sum_{s_t \in v_t} P(h_t) P(s_t|h_t) P(o|s_t) P(S_{t+1}|o, s_t) \quad [\text{Markov assumption}] \\ &= P(h_t) \sum_{s_t \in v_t} P(s_t|h_t) P(o|s_t) P(S_{t+1}|o, s_t) \\ &\propto \sum_{s_t \in v_t} P(s_t|h_t) P(o|s_t) P(S_{t+1}|o, s_t) \end{aligned} \quad (3.6)$$

The term $P(S_{t+1}|o, s_t)$ specifies the probability that the next state is S_{t+1} given that the previous state was $S_t = s_t$ and that o occurred. Let us consider a specific next state, $S_{t+1} = s'$. In the RUBRIC framework, a transition after o between state $S_t = s_t$ and state $S_{t+1} = s'$ is always mediated by some perceptual event e and some interpretation i . We capture this mediation using the Update Control’s function UPDATE-I:

$$s' = \text{UPDATE-I}(s_t, o, e, i) \quad (3.7)$$

In general, however, even assuming a specific state $S_t = s_t$, the agent may still face uncertainty about which perceptual event e_j and which interpretation i_l is correct for o . Moreover, we allow that the UPDATE-I function may be many-to-one: the same successor state s' may generally be reached through multiple interpretations $(e_1, i_1), \dots, (e_n, i_n)$ of o . Therefore, we have:

$$\begin{aligned} P(S_{t+1}|o, s_t) &= \sum_{E, I: S_{t+1} = \text{UPDATE-I}(s_t, o, E, I)} P(E, I|o, s_t) \\ &= \sum_{E, I: S_{t+1} = \text{UPDATE-I}(s_t, o, E, I)} P(E|o, s_t) P(I|E, o, s_t) \end{aligned} \quad (3.8)$$

This expression relates the agent’s uncertainty about which state succeeds s_t to the uncertainty its

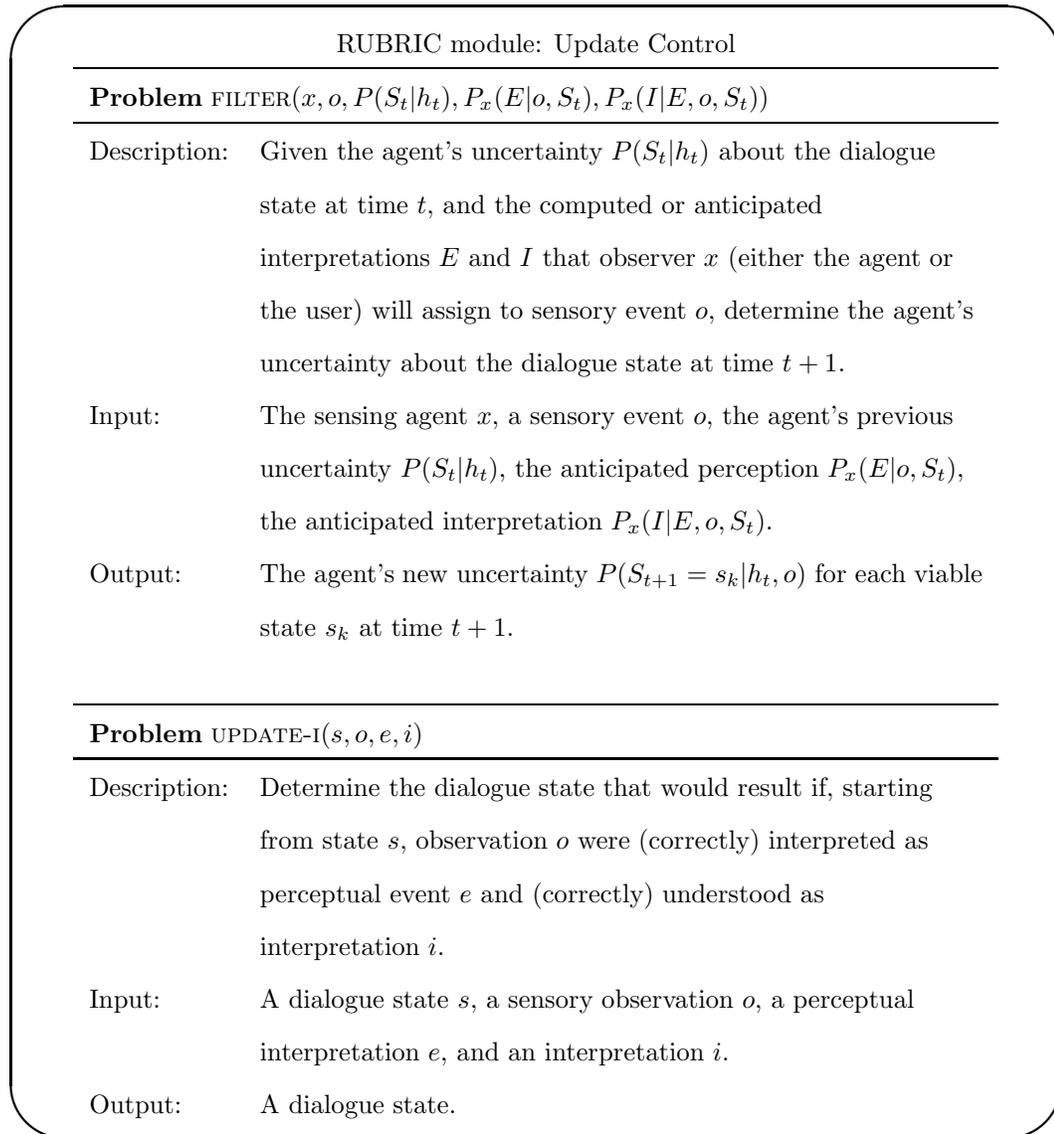


Figure 3.23: The Update Control module.

perception and interpretation algorithms report at s_t . Substituting (3.8) into (3.6) we have:

$$P(S_{t+1}, o, h_t) \propto \sum_{s_t \in v_t} P(s_t|h_t) P(o|s_t) \sum_{E,I: S_{t+1}=\text{UPDATE-I}(s_t,o,E,I)} P(E|o, s_t) P(I|E, o, s_t) \quad (3.9)$$

Substituting (3.9) into (3.5), we arrive at the final filtering operation carried out by the Update Control module's FILTER function:

$$P(S_{t+1}|h_{t+1}) \propto \sum_{s_t \in v_t} \underbrace{P(s_t|h_t)}_{\text{prior}} \underbrace{P(o|s_t)}_{\text{observation model}} \sum_{E,I: S_{t+1}=\text{UPDATE-I}(s_t,o,E,I)} \underbrace{P(E|o, s_t)}_{\text{perception}} \underbrace{P(I|E, o, s_t)}_{\text{interpretation}} \quad (3.10)$$

Thus, the agent's new uncertainty about which dialogue state it is in reflects its prior uncertainty, an observation model, its perceptual uncertainty, and its interpretation uncertainty.

3.3.8.1 COREF's Update Control

COREF's Update Control module is summarized in Figure 3.24.

UPDATE-A: The UPDATE-A function, which was introduced as (3.3) on page 91, captures all the effects associated with agent x taking a single action a starting from state s . In COREF, these effects are:

1. The state is updated to reflect the direct effects of x having done a . (These effects are hand-authored for each action type.)
2. A record is added to the history that x has performed action a .
3. All tasks on the task stack are updated to reflect x having done a . This may result in x does a state transitions occurring within the graph tasks.
4. Any now-completed tasks are automatically popped off the task stack.

UPDATE-I, UPDATE-P: As discussed in Section 3.3.6.1 (page 105), in COREF, an intentional interpretation takes the form $i = \langle x_1 : a_1, \dots, x_n : a_n \rangle$, where $n \geq 1$. For intentional interpretations of motor actions, COREF's implementation of UPDATE-I(s, o, e, i) simply executes the updates associated with $x_1 : a_1$ to $x_n : a_n$, in chronological order, and returns the resulting state. The observation

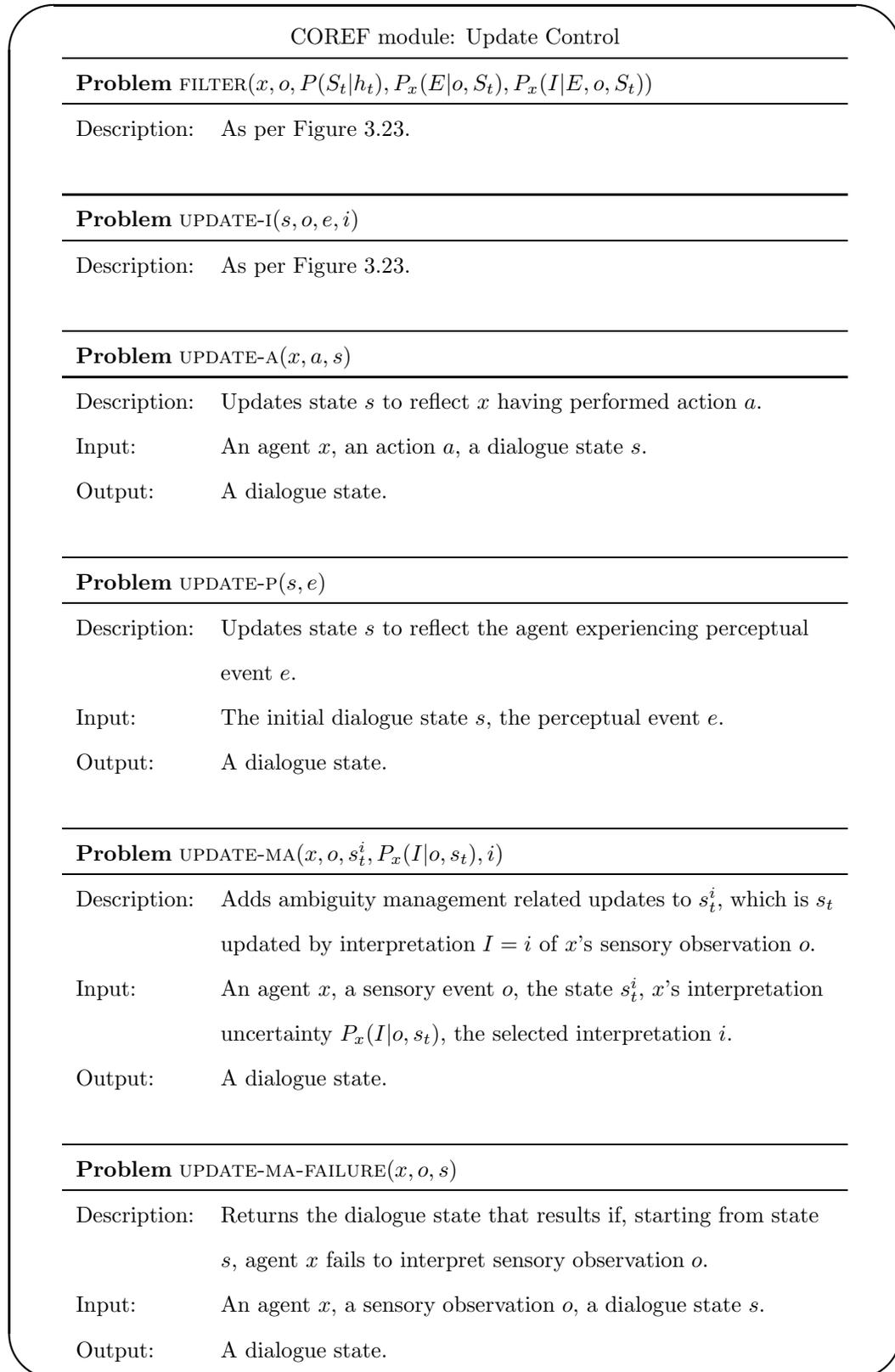


Figure 3.24: COREF's Update Control module.

o and perceptual event e are ignored. Formally, COREF implements $\text{UPDATE-I}(s, o, e, i)$ as follows:

$$\begin{aligned} \text{UPDATE-I}(s, o, e, i) &= \text{UPDATE-I}(s, i) \\ &= \text{UPDATE-I}(s, \langle x_1 : a_1, \dots, x_n : a_n \rangle) \\ &= \begin{cases} \text{UPDATE-A}(x_1, a_1, s) & \text{if } n = 1 \\ \text{UPDATE-I}(\text{UPDATE-A}(x_1, a_1, s), \langle x_2 : a_2, \dots, x_n : a_n \rangle) & \text{otherwise} \end{cases} \end{aligned}$$

For perceptual events e that are not interpreted in terms of intentional actions, e.g. the perception of the current objects in the game, an interpretation takes the form $i = \langle \text{COREF} : \text{perceive}(e) \rangle$. Times 6 and 7 in Figure 3.3 provide examples of this form. (Note that we use the $\text{COREF} : \text{perceive}(e)$ notation merely for uniformity in our formalism for interpretations; we do not mean to imply that perception should be thought of as an “action”.) COREF uses hand-built rules to update the dialogue state after non-intentional events, which we capture here with a function UPDATE-P :

$$\begin{aligned} \text{UPDATE-I}(s, o, e, i) &= \text{UPDATE-I}(s, \langle \text{COREF} : \text{perceive}(e) \rangle) \\ &= \text{UPDATE-P}(s, e) \end{aligned}$$

FILTER, UPDATE-MA, UPDATE-MA-FAILURE: COREF implements the filtering update presented in Section 3.3.8 under several simplifications. First, COREF does not exploit a substantive observation model $P(o|S_t)$. Instead, COREF assumes observations and dialogue states are independent, i.e. $P(o|S_t) = P(o)$. Under this assumption, the term $P(o|s_t)$ is a constant factor and can be removed from equation (3.10).

Further, since COREF does not face perceptual uncertainty, we identify the agent’s observation o with its perceptual event E . We thus have $E = o$, and $P(E|o, s_t) = 1$. The term $P(E|o, s_t)$ can therefore also be removed from equation (3.10). This leaves us with:

$$P(S_{t+1}|h_{t+1}) \propto \sum_{s_t \in v_t} \underbrace{P(s_t|h_t)}_{\text{prior}} \sum_{E, I: S_{t+1} = \text{UPDATE-I}(s_t, o, E, I)} \underbrace{P(I|o, s_t)}_{\text{interpretation}} \quad (3.11)$$

Finally, in COREF, the function UPDATE-I is one-to-one rather than many-to-one. This is because every dialogue state in COREF contains a historical record of all the actions that have previously occurred, including the order in which they occurred, and, further, COREF’s sensory event interpreter never hypothesizes multiple interpretations that contain exactly the same action sequences. The result is that, for any initial dialogue state $S_t = s_t$, each interpretation $I = i$ for which $P(I = i|o, s_t) > 0$ determines a unique successor state $s_t^i = \text{UPDATE-I}(s_t, o, e, i)$. This allows us to

eliminate the summation over alternative interpretations in equation (3.11):

$$P(S_{t+1} = s_t^i | h_{t+1}) \propto \sum_{s_t \in v_t} \underbrace{P(s_t | h_t)}_{\text{prior}} \underbrace{P(I = i | o, s_t)}_{\text{interpretation}} \quad (3.12)$$

This, then, is the probability that COREF's FILTER function actually computes. Generally, each state s_t^i for which $P(S_{t+1} = s_t^i | h_{t+1}) > 0$ corresponds to a viable successor state at time $t + 1$. The exact successor state that remains is the result of adding several book-keeping and uncertainty management related updates to s_t^i . We capture these updates using a function,

$$\text{UPDATE-MA}(x, o, s_t^i, P_x(I|o, s_t), i),$$

which performs the following updates to s_t^i :

1. push a $\text{ManageAmbiguity}(x, o)$ task onto the stack, where x is the agent who is interpreting their observation o ;
2. store a record of the interpretation uncertainty $P_x(I|o, s_t)$ associated with x 's interpretation of o .
3. store a record that this particular state selects interpretation $I = i$.

The idea COREF implements, here, is that some interpretation $I = i$ is the correct (intended) interpretation, and that correct interpretation i determines which successor dialogue state is the correct one. However, no matter which successor dialogue state is the correct one, the observer x may coherently follow up their observation o with ambiguity-managing actions (update 1). Keeping the record $P_x(I|o, s_t)$ of the alternative interpretations as part of the dialogue state makes those ambiguities available in subsequent interpretation (update 2). Finally, keeping a record $I = i$ makes it possible to distinguish which interpretation is active in each hypothetical successor state (update 3).

For each state s_t^i for which $P(S_{t+1} = s_t^i | h_{t+1}) > 0$, COREF computes $u(s_t^i)$ as a possible successor state, where:

$$u(s_t^i) = \text{UPDATE-MA}(x, o, s_t^i, P_x(I|o, s_t), i).$$

The probabilities COREF's FILTER function assigns to successor states are then:

$$P(S_{t+1} = u(s_t^i) | h_{t+1}) \leftarrow P(S_{t+1} = s_t^i | h_{t+1})$$

COREF’s FILTER function performs a few additional steps. To keep search tractable for real-time interaction, COREF tracks a maximum of 3 states. If more than 3 states are possible, the 3 most probable are retained, and the others discarded. Further, after each object is completed (by the director taking a public `continueTask(T)` action, or either agent taking a public `skip(T)` action), COREF discards all but the most probable state, to avoid retaining unilluminating historical ambiguities. If either of these adjustments is made, the probability mass is renormalized across the remaining viable states. An example of a dropped state occurs at time 6 in Figure 3.3, when COREF drops state `s2704` after continuing on to the next object.

Finally, there is a special case when a sensory observation has no interpretations. In this case, a set of possible successor states cannot be determined as above. Instead, COREF translates each previously viable state $S_t = s_t$ into a new viable state by performing book-keeping and uncertainty management updates similar to the above. In particular, when x finds no interpretations for their observation o , the function `UPDATE-MA-FAILURE(x, o, s_t)` performs the following updates on each state s_t for which $P(S_t = s_t|h_t) > 0$:

1. push a `ManageAmbiguity(x, o)` task onto the stack, where x is the agent who is interpreting their observation o ;
2. store a record that x found no interpretations for o .

The idea here is to make it coherent for x to follow up their interpretation failure with ambiguity management actions (update 1), as usual, and to note the presence of an interpretation failure in the state (update 2). In terms of probabilities, we assume an interpretation failure is not informative with respect to the correct dialogue state, and simply maintain an identical distribution on the isomorphic successor states. Formally, if we let $f(x, o, s_t) = \text{UPDATE-MA-FAILURE}(x, o, s_t)$, then COREF’s FILTER function assigns the following probabilities in the case of an interpretation failure:

$$P(S_{t+1} = f(x, o, s_t)|h_{t+1}) \leftarrow P(S_t = s_t|h_t)$$

3.3.9 Intention Generator

The RUBRIC Intention Generator module is summarized in Figure 3.25. The module is responsible for generating contributive actions for the agent to perform as part of its interaction with the user. This problem-solving involves identifying a motor action sequence that the agent finds acceptable given its current uncertainty about the dialogue state, and given its best estimate of how the user will interpret the public evidence that its actions will provide.

RUBRIC module: Intention Generator	
Problem: GENERATE($P(S_t h_t)$, a, u ; $\mathbf{G}, \mathbf{UC}, \mathbf{SEI}$)	
Description:	Generates a contributive action for agent a to perform with respect to the user u .
Input (event):	The agent a , the user u , A 's uncertainty the $P(S_t h_t)$.
Input (resources):	The grammar \mathbf{G} , the update control \mathbf{UC} , and the sensory event interpreter \mathbf{SEI} .
Output:	A generatum of the form $\langle m^*, o_u, P_u(E = e_j o_u, S_t = s_k), P_u(I = i_l E = e_j, o_u, S_t = s_k) \rangle$ where o_u is the user's anticipated sensory observation of motor action sequence m^* , $P_u(E = e_j o_u, S_t = s_k)$ is the user's anticipated perceptual classification of o_u , and $P_u(I = i_l E = e_j, o_u, S_t = s_k)$ is the user's anticipated interpretation of o_u .

Figure 3.25: The Intention Generator module.

As with the Sensory Event Interpreter module (Section 3.3.6), the RUBRIC Intention Generator module not reflect any substantive assumptions about the form that interpretations of actions take. Rather, it aims only to situate the reasoning an agent performs to identify a contributive action within a coherent probabilistic framework. To do so, it assumes that the Sensory Event Interpreter module is available as a resource for generation. This allows the generation process to try to anticipate the interpretations that the user will assign to an utterance, and crucially, use those anticipated interpretations to guide generation decisions. By choosing actions that the user can be expected to interpret in a specific way, the agent can identify a contributive action to perform.

The output of the GENERATE function is a generatum g , where

$$g = \langle m^*, o_u, P_u(E = e_j|o_u, S_t = s_k), P_u(I = i_l|E = e_j, o_u, S_t = s_k) \rangle.$$

This output format includes the generated motor action sequence m^* and also the user's anticipated sensory observation o_u , perceptual classification of that observation $P_u(E = e_j|o_u, S_t = s_k)$, and interpretation $P_u(I = i_l|E = e_j, o_u, S_t = s_k)$. As shown in Figure 3.4 (page 84), a RUBRIC agent

uses the *user's* anticipated interpretations to filter its *own* uncertainty about which dialogue state it is in, after it makes an utterance. This enforces a hard constraint, after an agent action, that keeps the agent's estimate of the dialogue state in line with the evidence it believes its action has actually provided to the user. It also makes it possible for an agent to implement, if it anticipates that its own utterance or motor action is ambiguous, an underspecified update to the dialogue state. We will see momentarily that COREF sometimes performs such underspecified updates following its utterances.

3.3.9.1 COREF's Intention Generator

COREF's Intention Generator represents action sequences using the same representation scheme that COREF's Sensory Event Interpreter uses to assign interpretations to sensory events (see Section 3.3.6.1). The Intention Generator thus represents the contributions it might perform as action sequences of the form $a^* = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$, in which agent A_1 performs action a_1 , then agent A_2 performs action a_2 , and so on. The action sequences COREF entertains all terminate in a public motor action or dialogue act by COREF. The public action is preceded by zero or more tacit actions performed either by COREF or by COREF's human partner. We will use the words *interpretation*, *intention*, and *contribution* interchangeably in discussing these sequences.

COREF's generation process adapts established natural language generation techniques to a setting of task-oriented generation under uncertainty. Our basic building block is the SPUD generation algorithm of Stone et al. (2003). SPUD is a search-based generation algorithm that uses a lexicalized tree-adjoining grammar to incrementally construct an utterance that achieves a communicative goal (in COREF's case, a dialogue act) in a way that is unambiguous in a specific context. SPUD's reasoning about ambiguity is based on grammatically specified semantic constraints, which are solved in context to produce a set of contextually supported variable assignments. SPUD also uses a salience model, based on distractor sets (Dale and Reiter, 1995), which allows the algorithm to evaluate the degree of ambiguity associated with an intended interpretation for an utterance. Given an input context, SPUD delivers a *communicative intent*, which includes a sentence, a syntactic and semantic analysis of the sentence, and an intended interpretation (i.e. variable assignment) for the utterance's semantics (i.e. constraints). If SPUD's generation process succeeds, the selected communicative intent unambiguously achieves the communicative goal in the input context.

Because SPUD is designed to operate under certainty about the context in which the utterance occurs, it was necessary to assimilate SPUD into a broader framework that includes COREF's uncertainty about the context as well as COREF's reasoning about the alternative tacit actions

that may have occurred. COREF does so by invoking SPUD in one or more alternative contexts, with a communicative goal appropriate to each context individually, and by using the Sensory Event Interpreter to assess the interpretations the user might assign to the utterance in other relevant contexts. The specific contexts in which SPUD is invoked are identified using a horizon graph to circumscribe a set of relevant input contexts for SPUD (the *horizon*) as well as the coherent communicative goals that COREF might adopt in each context. The horizon graph representation is shared with COREF's Sensory Event Interpreter module; see Section 3.3.6.1.

In the next several sections, we will define and illustrate COREF's generation process using examples from the subdialogue in Figure 3.3.

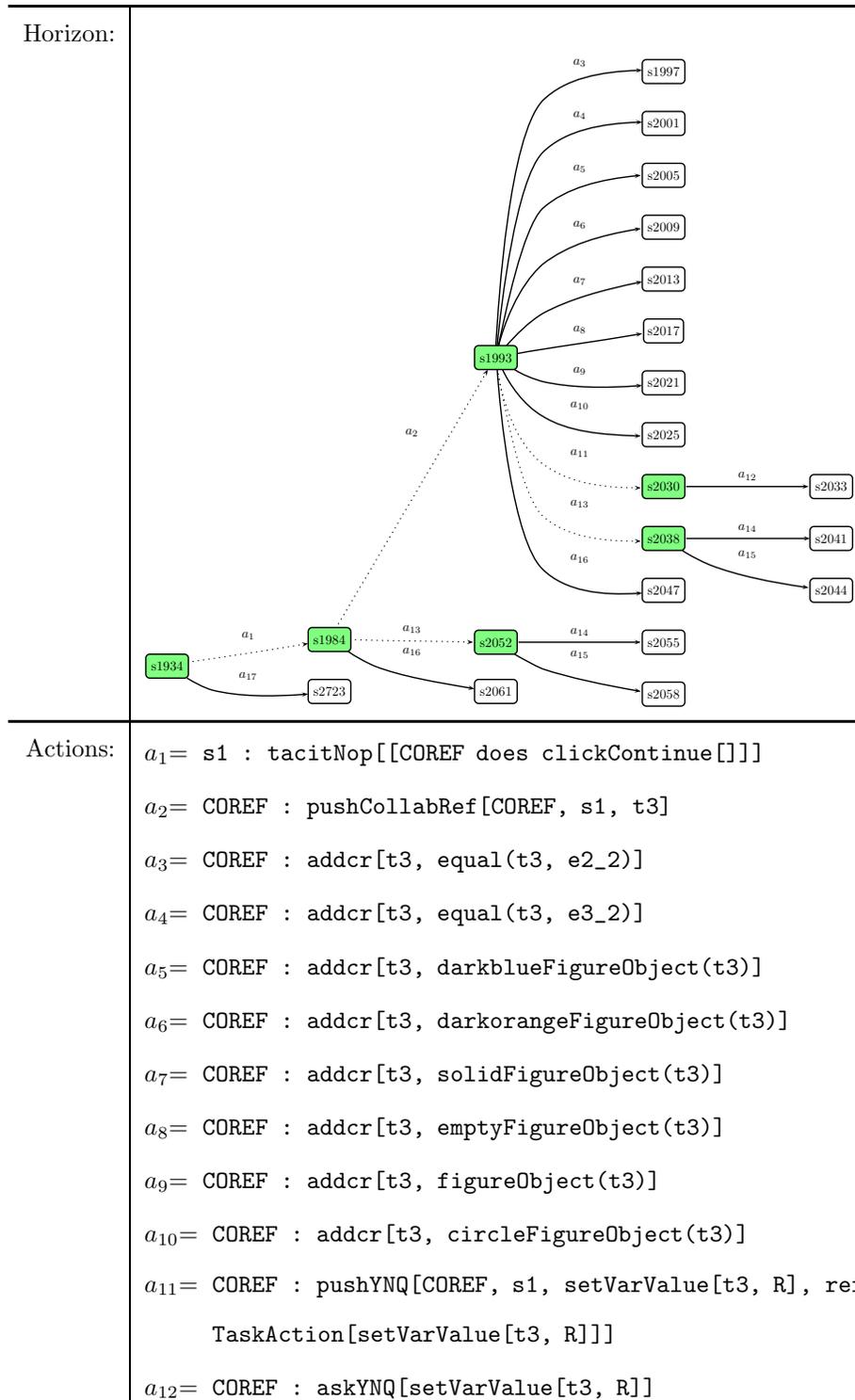
3.3.9.1.1 Generating a contributive utterance while certain about the context COREF begins the subdialogue in Figure 3.3 by saying *the blue circle*. To generate this utterance, COREF first constructs the horizon graph for the current dialogue state, s1934. This horizon graph is depicted in Figure 3.26. This graph shows which the various actions that could coherently be taken from state s1934.

For example, one action COREF could take is a_{17} , in which COREF skips the current object. Another coherent action, a_1 , relates to the action that immediately preceded COREF's utterance of *the blue circle*. The previous action, which is not included in Figure 3.3, was that COREF performed the public action `clickContinue[]` to finish off the previous object (before the blue circle). Therefore, as usual, on the top of the stack in state s1934, there is a `ManageAmbiguity` task that allows s1 to follow up on any perceived ambiguity in COREF's previous action. This means that one tacit action that could coherently come next would be for s1 to tacitly decline to follow up on any uncertainty regarding COREF's prior `clickContinue[]` action. This tacit action is labeled a_1 in the figure. We allow COREF to implicate that s1 has performed a_1 by constructing an intention that includes a_1 . After a_1 , COREF has additional options for acting coherently, including initiating collaborative reference to the next target object (action a_2), pushing a reminder task regarding the previous object (a_{13}), and other options.

Figure 3.26: Possible horizon graph for COREF: *the blue circle*

continues on next page...

Figure 3.26: Possible horizon graph for COREF: *the blue circle*
(continued)



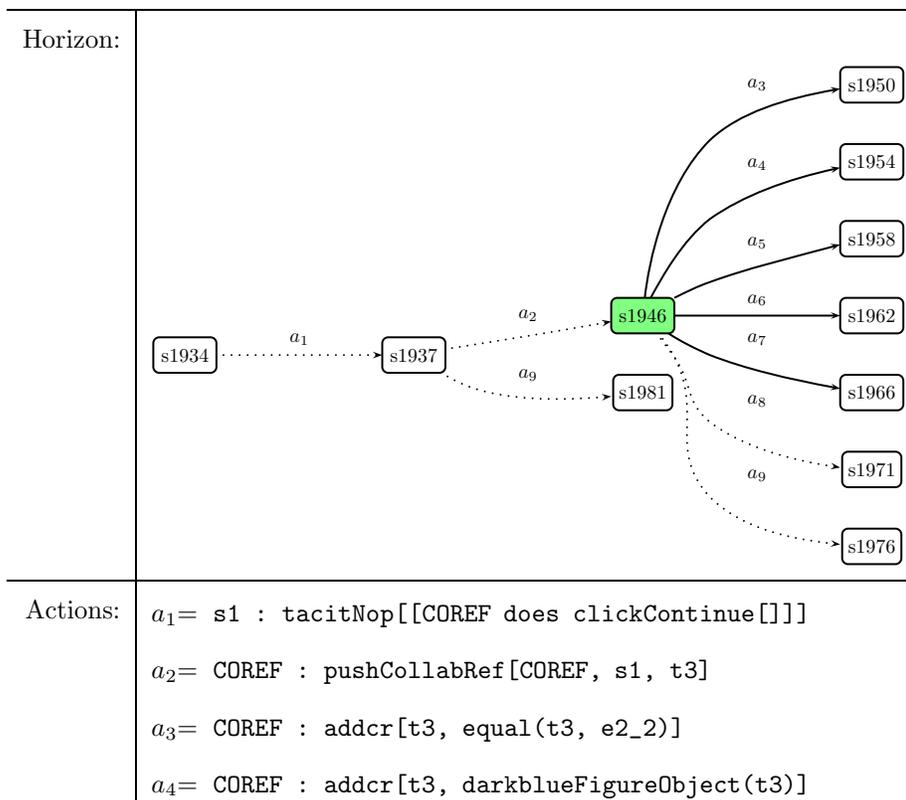
continues on next page...

Figure 3.26: Possible horizon graph for COREF: *the blue circle*
(continued)

a_{13} = COREF : pushRemind[COREF, s1, past, refuseTaskAction, addToScene[rh1_1]]
a_{14} = COREF : askYNQ[past[s1, addToScene[rh1_1]]]
a_{15} = COREF : command[s1, addToScene[rh1_1]]
a_{16} = COREF : past[s1, addToScene[rh1_1]]
a_{17} = COREF : skip[t3]

In general, because this figure circumscribes the actions COREF *could* do (or implicate), rather than what it is *willing* to do, we describe Figure 3.3 as a *possible horizon graph*. COREF currently uses a set of hand-built rules that decide which of the possible actions the agent is willing to do. We present the corresponding acceptable actions, for this example, in the *acceptable horizon graph* of Figure 3.27.

Figure 3.27: Acceptable horizon graph for COREF: *the blue circle*



continues on next page...

Figure 3.27: Acceptable horizon graph for COREF: *the blue circle*
(continued)

```

a5= COREF : addcr[t3, solidFigureObject(t3)]
a6= COREF : addcr[t3, figureObject(t3)]
a7= COREF : addcr[t3, circleFigureObject(t3)]
a8= COREF : pushYNQ[COREF, s1, setVarValue[t3, R], refuse-
      TaskAction[setVarValue[t3, R]]]
a9= COREF : pushRemind[COREF, s1, past, refuseTaskAction,
      addToScene[rh1_1]]

```

The acceptable horizon graph eliminates various actions that COREF is unwilling to do. For example, COREF is never willing to skip objects; action a_{17} is therefore not present in the acceptable horizon graph. COREF is willing to initiate collaborative reference to the next target object (a_2), and to perform the `addcr` dialogue act to contribute a constraint that will help the matcher identify the target object, $t3$ (a_3 - a_{10}). However, as director, COREF is only willing to assert constraints that it knows are true of the target object. For example, while the possible horizon graph makes it coherent for COREF to perform action `addcr[t3, equal(t3, e3_2)]`, which identifies the target object with $e3_2$ (the empty orange circle), COREF knows that the target object is actually $e2_2$, and so does not find this action acceptable. In general, COREF's rules for the acceptability of actions allow it to reject each possible action or to accept it under 1 or more refined variable assignments.

In generation, COREF constructs both the possible and acceptable horizon graphs for the dialogue states it thinks it might be in. Crucially, while the acceptable horizon graph circumscribes the communicative goals COREF is willing to adopt, it is the *possible* horizon graph that is used in anticipating user interpretations. This reflects an assumption that COREF's interlocutor, $s1$, also knows what the possible horizon graph is for each dialogue state. For example, looking at the possible horizon graph in Figure 3.26, from state $s1934$, COREF expects that $s1$, the matcher, will view both $a_3 = \text{COREF} : \text{addcr}[t3, \text{equal}(t3, e2_2)]$ and $a_4 = \text{COREF} : \text{addcr}[t3, \text{equal}(t3, e3_2)]$ as coherent dialogue acts that COREF might make next (after several tacit actions).²⁰

COREF uses the alternative coherent dialogue acts in the possible horizon graph to help it generate contributive output utterances using SPUD. Specifically, COREF first identifies an acceptable tacit action sequence that leads to a horizon state from which a dialogue act can be performed, such

²⁰For example, COREF might perform the first by uttering *the blue circle*, or the second by uttering *the empty orange circle*.

as s1946 in the acceptable horizon graph in Figure 3.27.²¹ It then selects an acceptable dialogue act, such as $a_3 = \text{addcr}[\text{t3}, \text{equal}(\text{t3}, \text{e2_2})]$ in Figure 3.27. It then returns to the possible horizon graph, and uses it to invoke SPUD to try to generate an utterance that achieves this communicative goal.

In general, the problem we would like for SPUD to solve is to identify not only an utterance that can express a specific dialogue act, such as a_3 , but also an utterance which distinguishes an intended horizon state in the possible horizon graph (Thomason et al., 2006), such as s1993 in Figure 3.26. To achieve this, it would perhaps be ideal to modify SPUD so that SPUD takes as input not a single context, but rather a set of alternative contexts, or perhaps a probability distribution over alternative contexts. COREF instead exploits a heuristic approach to try to allow the off-the-shelf SPUD algorithm to identify an appropriate utterance under uncertainty about the context. On this approach, in order to try to capture the ambiguities in the possible horizon, rather than passing a single context like s1993 directly to SPUD, COREF instead packages up the possible horizon as a *pseudo-context*, and passes the pseudo-context as input to SPUD. In this particular example, COREF constructs a pseudo-context equal to $Z(s1934) = \{s1934, s1984, s1993, s2030, s2038, s2052\}$. Conceptually, these are all the dialogue states (contexts) in which COREF’s addressee may try to interpret the public utterance.

A pseudo-context $\mathcal{C} = Z(s)$ behaves as follows. The set of assignments $\Sigma = \{\sigma_1, \dots, \sigma_n\}$ that satisfy an utterance’s semantic constraints in \mathcal{C} is the union $\Sigma = \Sigma_1 \cup \dots \cup \Sigma_k$, where Σ_i is the set of assignments that solve the utterance’s presuppositions and dialogue acts in state $s_i \in Z(s)$. The effect of this behavior is to increase the amount of ambiguity that is visible to SPUD beyond what would be present in a single context. In particular, the ambiguities are broadened to reflect ambiguities arising throughout the possible horizon. Ideally, this additional ambiguity would sometimes lead SPUD to slightly overspecify its output utterances, beyond what would be necessary in a single context, in order to distinguish the intended contribution from distractor contributions in the possible horizon. However, to date, we have not investigated this hunch quantitatively.

In COREF, generation occurs in two steps. First, SPUD is invoked in an attempt to express an acceptable dialogue act, using the possible horizon as a pseudo context. Second, if SPUD has produced a successful utterance, COREF’s intention generator interprets the utterance using the Sensory Event Interpreter module. (Note that the Sensory Event Interpreter evaluates the utterance’s semantic constraints on a context-by-context basis; no pseudo-context is involved.) If all of the

²¹Be aware that, due to the method of preparation for these figures, the state numbers and action numbers are not generally shared between these figures for the possible and acceptable horizon graphs, even in cases where the actions and states are identical.

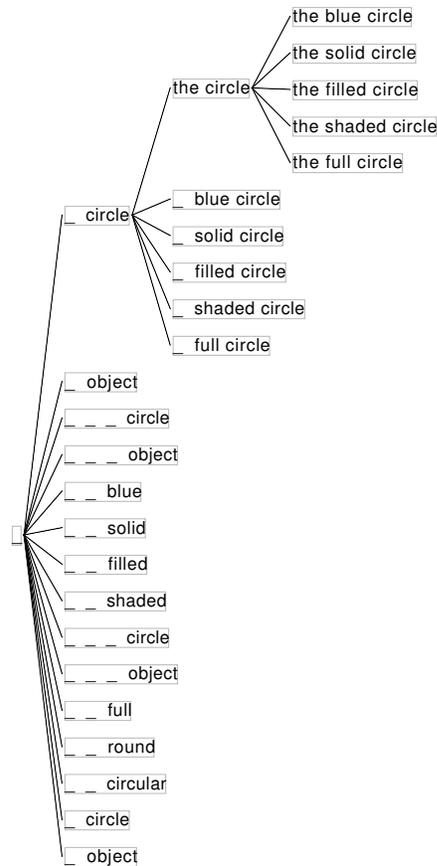


Figure 3.28: SPUD generates *the blue circle* for COREF. In this figure, each search node considered by SPUD is depicted using only the surface text at that search node. Nodes are ranked vertically, with higher ranked nodes above lower ranked nodes. An underscore indicates the presence of a syntactic gap in the provisional syntactic structure at the node.

actions that appear in all the interpretations found by the Sensory Event Interpreter are acceptable to COREF, then COREF accepts the output utterance. In the case that multiple interpretations are supported by an utterance, if COREF accepts the utterance, we view COREF as *willing to be interpreted as making any of those contributions*. We refer to this scenario as one in which COREF makes an underspecified contribution.

If one or more of the interpretations for an output utterance is unacceptable to COREF, it tries reinvoking SPUD, either with a different communicative goal, or from a different horizon state. This process repeats until an acceptable utterance is found, or until all communicative options are exhausted.

In this particular example, when SPUD is invoked in pseudo-context $Z(s1934)$, with the communicative goal of performing dialogue act $\text{addcr}[t3, \text{equal}(t3, e2_2)]$, SPUD generates *the blue*

circle. The search process by which SPUD selects this utterance is illustrated in Figure 3.28. In generating this utterance, SPUD decides on *the blue circle* rather than simply *the circle* because the semantic constraints associated with *the circle* support two interpretations: a_3 and a_4 in the possible horizon graph of Figure 3.26. These two interpretations correspond to the two circles that are visible as pending objects in the experiment interface. Because the ambiguity is exposed in the coherent dialogue acts in the possible horizon, SPUD determines that *the circle* would be ambiguous, and proceeds to generate *the blue circle*, which uniquely distinguishes COREF’s requested dialogue act a_3 in the pseudo-context. SPUD’s rejection of contextually ambiguous utterances like *the circle* is part of how COREF identifies contributive utterances in context.

After SPUD returns a successful utterance, COREF invokes its Sensory Event Interpreter to try to anticipate all the various interpretations the user might assign, across all the different dialogue states the agent might be in. In this case, COREF’s Sensory Event Interpreter module reports that *the blue circle* has only a single interpretation, $i_{1,1}$:

$$i_{1,1} = \langle s1 : \text{tacitNop}[[\text{COREF does clickContinue}[]]],$$

$$\text{COREF} : \text{pushCollabRef}[\text{COREF}, s1, t3],$$

$$\text{COREF} : \text{addcr}[t3, \text{equal}(t3, e2_2)],$$

$$\text{COREF} : \text{setPrag}[\text{inFocus}(Z), \text{inFocus}(t3)] \rangle$$

By uttering *the blue circle*, COREF therefore anticipates that it would be recognizably contributing that $s1$ has declined to manage any perceived uncertainty regarding COREF’s previous utterance (by implication), that COREF has initiated collaborative reference to the next target object (by implication), that COREF has added an equality constraint for the next target object (a dialogue act), and that COREF has put the next target object into focus (a dialogue act).

Since COREF finds all these component actions acceptable, COREF’s Intention Generator accepts *the blue circle* as its contributive utterance and returns action sequence

$$m^* = \langle \text{say}[\text{the blue circle}] \rangle$$

along with the Sensory Event Interpreter’s anticipated user interpretation $i_{1,1}$.

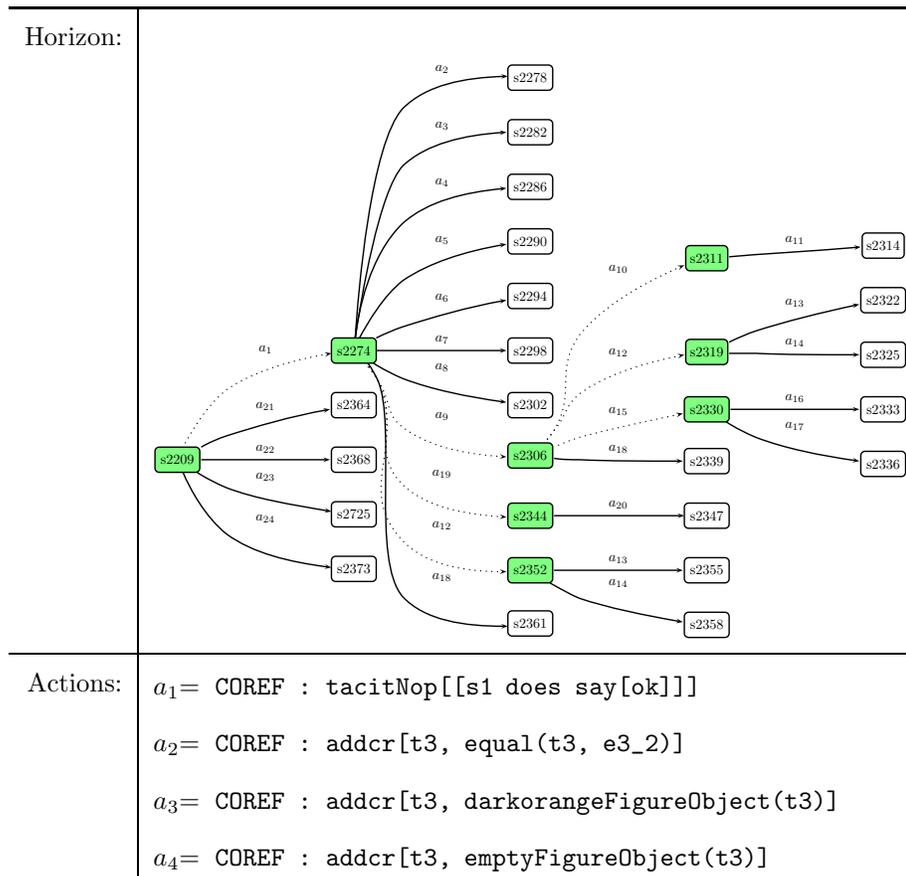
3.3.9.1.2 Generating a contributive utterance under uncertainty about the context

We now use COREF’s utterance *did you add it?* at time 3 in the dialogue of Figure 3.3 to illustrate how COREF formulates a contributive utterance under uncertainty about the current context.

When COREF faces uncertainty about the context, it first tries to generate a contribution that

is coherent in the most probable context. At time 3, s1 has just said *ok*, and COREF views s2209, in which *ok* simply contributed a *nop*, as the most probable dialogue state. Figure 3.29 shows COREF’s acceptable horizon graph for state s2209. In state s2209, even though COREF is uncertain what s1’s utterance of *ok* meant, COREF is willing to tacitly decline the opportunity to clarify — perhaps it could do so by uttering something like *what do you mean ok?* — by performing a *tacitNop* action (a_1). Moving forward, its rules for action acceptability allow it to implicate, on the grounds that COREF’s utterance *the blue circle* added a constraint that uniquely identified the target object, that the user has identified the target object. COREF’s domain model captures the user’s mental identification of the target object as action $a_9 = s1 : \text{setVarValue}[\text{t3}, \text{e2_2}]$. The effect of this tacit mental action is to make it part of the context that target *t3* is object *e2_2*. COREF’s acceptability rules further allow the agent to push a *Remind* task (a_{15}), which allows COREF to coherently ask whether s1 has added the object to their scene (a_{16}).

Figure 3.29: Acceptable horizon graph for COREF: *did you add it?*



continues on next page...

Figure 3.29: Acceptable horizon graph for COREF: *did you add it?*

(continued)

```

a5= COREF : addcr[t3, darkblueFigureObject(t3)]
a6= COREF : addcr[t3, figureObject(t3)]
a7= COREF : addcr[t3, circleFigureObject(t3)]
a8= COREF : addcr[t3, solidFigureObject(t3)]
a9= s1 : setVarValue[t3, e2_2]
a10= COREF : pushYNQ[COREF, s1, addToScene[e2_2], refuse-
    TaskAction[addToScene[e2_2]]]
a11= COREF : askYNQ[addToScene[e2_2]]
a12= COREF : pushRemind[COREF, s1, past, refuseTaskAction,
    addToScene[rh1_1]]
a13= COREF : askYNQ[past[s1, addToScene[rh1_1]]]
a14= COREF : command[s1, addToScene[rh1_1]]
a15= COREF : pushRemind[COREF, s1, past, refuseTaskAction,
    addToScene[e2_2]]
a16= COREF : askYNQ[past[s1, addToScene[e2_2]]]
a17= COREF : command[s1, addToScene[e2_2]]
a18= COREF : past[s1, addToScene[rh1_1]]
a19= COREF : pushYNQ[COREF, s1, setVarValue[t3, R], refuse-
    TaskAction[setVarValue[t3, R]]]
a20= COREF : askYNQ[setVarValue[t3, R]]
a21= COREF : nop[[s1 does say[ok]]]
a22= COREF : flagProblematic[[s1 does say[ok]]]
a23= COREF : skip[t3]
a24= COREF : abandonTasks[1]

```

COREF therefore adopts as its communicative goal to perform the dialogue act a_{16} :

$$a_{16} = \text{COREF} : \text{askYNQ}[\text{past}[\text{s1}, \text{addToScene}[\text{e2_2}]]].$$

The situation differs from the generation of *the blue circle*, however, in that COREF is currently

uncertain which of two dialogue states, s2209 or s2199, is the correct dialogue state. The possible horizon graphs for these two states are given in Figures 3.30 and 3.31, respectively.

Figure 3.30: Possible horizon graph from state s2209 for COREF:

did you add it?

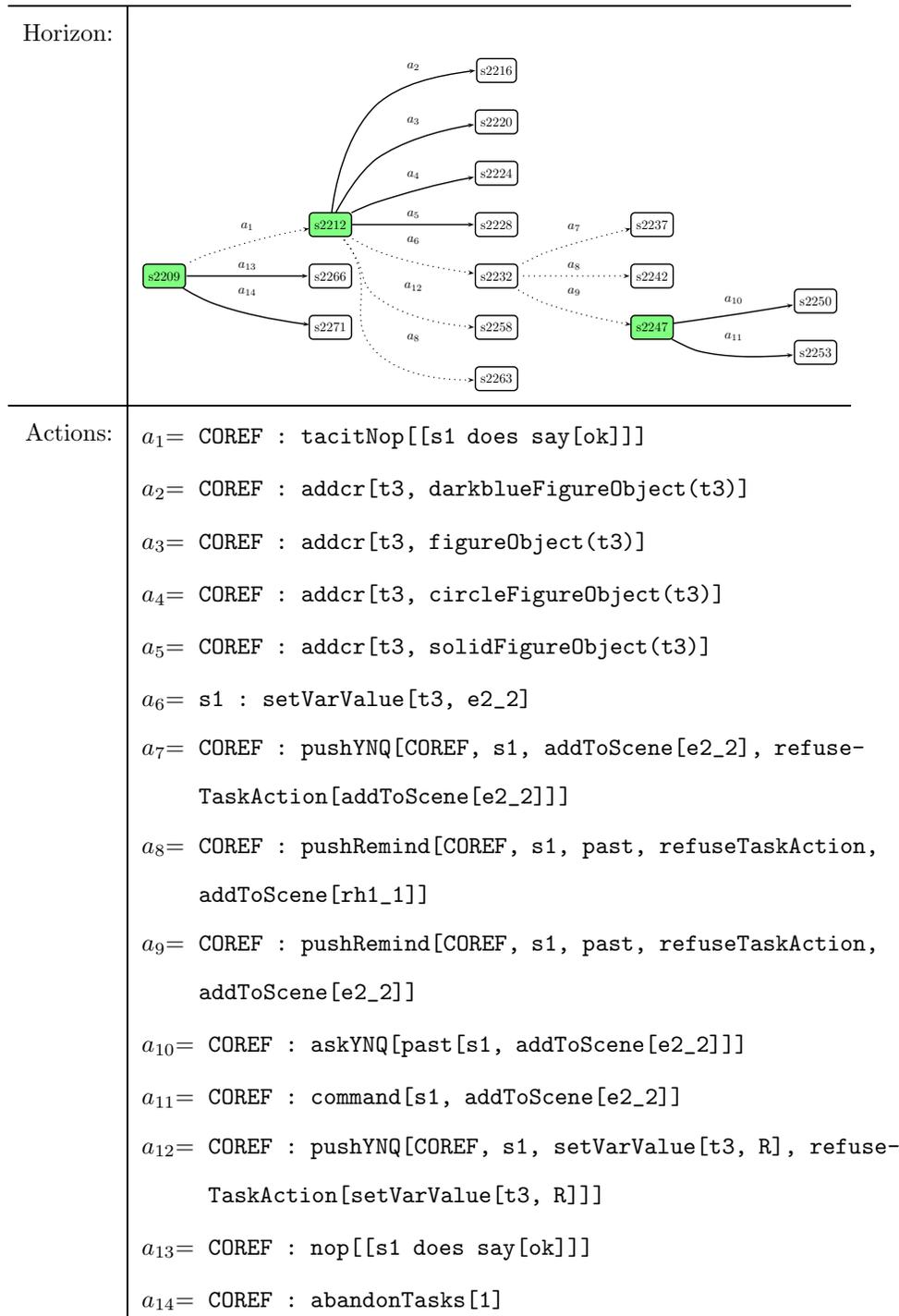
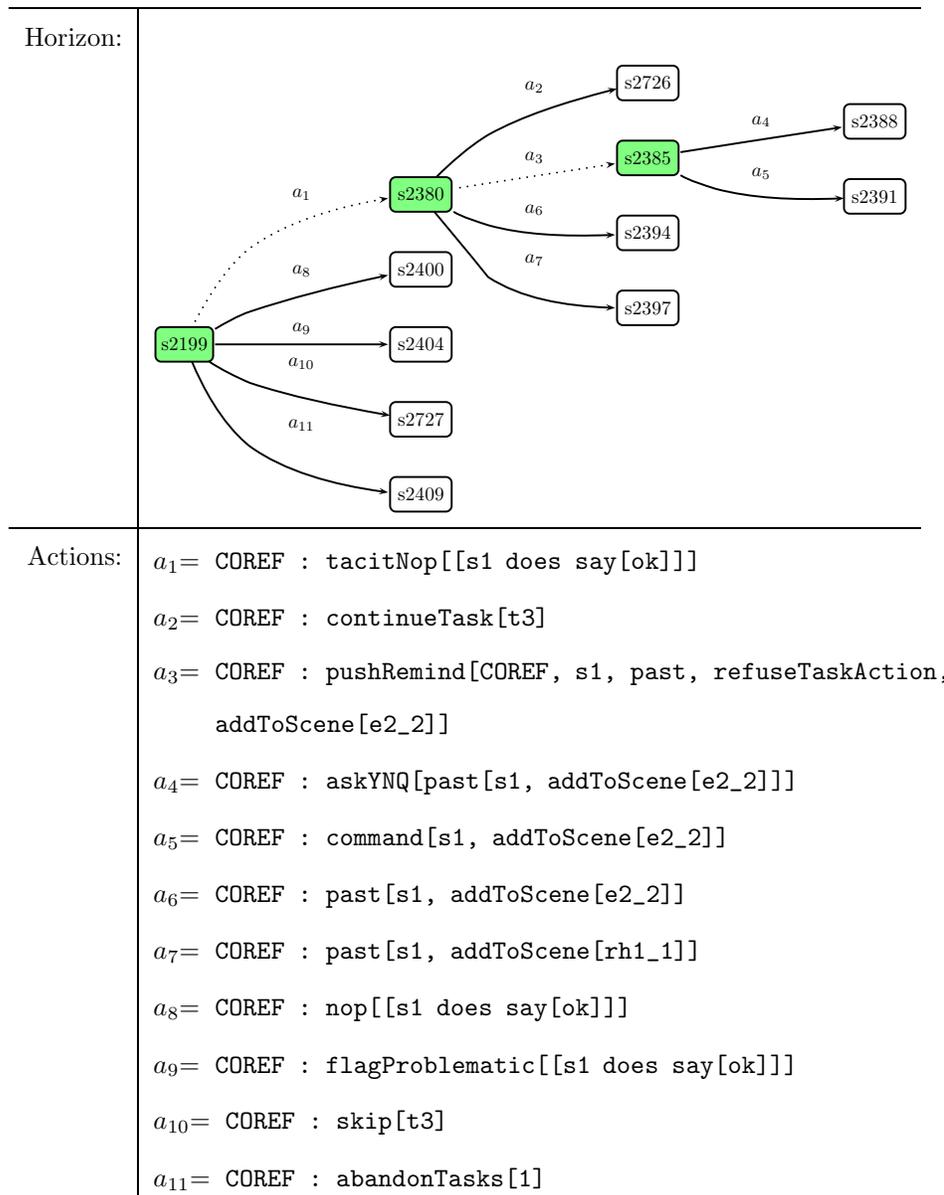


Figure 3.31: Possible horizon graph from state s2199 for COREF:

did you add it?

These figures show that COREF has a number of coherent options in each of these two states. However, note that COREF's options differ in the two states. In particular, in state s2199, but not in state s2209, the human matcher s1 has already contributed by way of intention $i_{2,1}$ that they have identified the object and added it to their scene. If the true state is s2199, therefore, COREF cannot, according to its task models, coherently implicate that s1 has identified the object — s1 has already made that contribution, and COREF's task models do not generally support a repetition of previous contributions. In state s2209, on the other hand, where s1's *ok* merely contributed

a `nop`, COREF’s task models *do* support support COREF implicating that `s1` has identified the object. Further, because `s1` has not yet contributed that they have identified the object, COREF can make a number of additional coherent contributions, including adding additional constraints about the target object. An important limitation, however, is that from state `s2209`, the only way that COREF can coherently push a `Remind` task for `e2_2` (a_9), which allows COREF to coherently ask the user if they added that object, is if COREF first implicates that the user has identified the object. This constraint is enforced through a rule that prevents an agent from coherently initiating a `Remind` task about an action unless that action is the next action the other agent needs to do. (Intuitively, this implements a heuristic that, in a task-oriented collaboration like COREF’s object identification game, you shouldn’t remind someone to do something if it isn’t the next thing they need to do.)

From COREF’s perspective, either of these two states might be the true state, and further, its human interlocutor `s1` might be assuming that either of these states is the true state. In this situation, in order to invoke SPUD, COREF anticipates interpretations in both states by constructing an expanded pseudo-state $\mathcal{C} = Z(s2209) \cup Z(s2199)$. (The horizons $Z(s2209)$ and $Z(s2199)$ are shaded in green in Figures 3.30 and 3.31, respectively.) Given this pseudo-state, and the communicative goal of performing COREF’s chosen dialogue act, `COREF : askYNQ[past[s1, addToScene[e2_2]]]`, SPUD identifies *did you add it?* as an unambiguous utterance.

COREF then invokes its Sensory Event Interpreter to anticipate how the user will interpret an utterance of *did you add it?* The Sensory Event Interpreter relies on the horizon graphs of Figures 3.30 and 3.31 to determine that *did you add it?* supports two interpretations, one from each of the current possible dialogue states. From `s2209`, in which `s1`’s previous *ok* simply contributed a `nop`, the utterance supports interpretation $i_{3,2}$:

$$i_{3,2} = \langle \text{COREF : tacitNop}[[\text{s1 does say}[\text{ok}]]], \\ \text{s1 : setVarValue}[\text{t3, e2_2}], \\ \text{COREF : pushRemind}[\text{COREF, s1, past, refuseTaskAction, addToScene}[\text{e2_2}]], \\ \text{COREF : askYNQ}[\text{past}[\text{s1, addToScene}[\text{e2_2}]]], \\ \text{COREF : setPrag}[\text{inFocus}(\text{Y}), \text{inFocus}(\text{e2_2})] \rangle$$

If the context were `s2209`, then by uttering *did you add it?*, COREF anticipates that it would be contributing that COREF is declining to manage any perceived uncertainty regarding `s1`’s utterance of *ok*, that `s1` has identified the target object (by implication), that COREF has initiated a reminder task (by implication), that COREF is asking a yes/no question as part of that reminder task, and

that COREF is putting object `e2_2` into focus.

From state `s2199`, in which `s1`'s previous `ok` contributed more, COREF's Sensory Event Interpreter reports that *did you add it?* supports interpretation $i_{3,1}$:

```
 $i_{3,1} = \langle$ 
COREF : tacitNop[[s1 does say[ok]]],
    COREF : pushRemind[COREF, s1, past, refuseTaskAction, addToScene[e2_2]],
    COREF : askYNQ[past[s1, addToScene[e2_2]]],
    COREF : setPrag[inFocus(Y), inFocus(e2_2)] $\rangle$ 
```

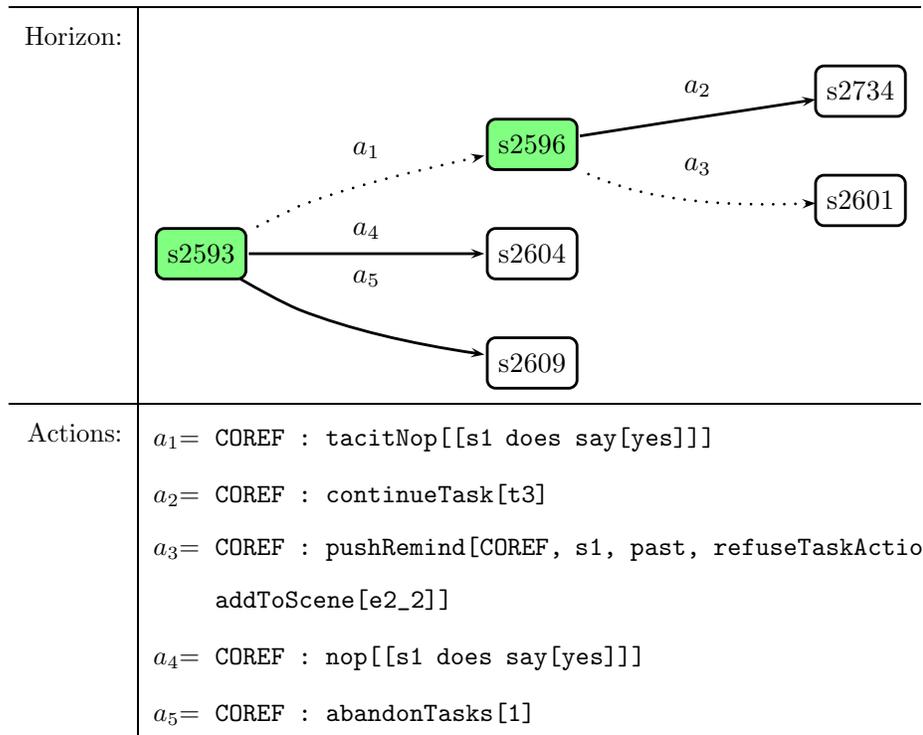
If the context were `s2199`, then by uttering *did you add it?*, COREF anticipates that it would be making a similar contribution, except now COREF would not be implicating that `s1` has identified the target object.

Because COREF finds both of these contributions acceptable, in the corresponding dialogue states, COREF's Intention Generator accepts *did you add it?* and returns action sequence $m^* = \langle \text{say}[\text{did you add it?}] \rangle$, along with the Sensory Event Interpreter's anticipated user interpretations. We view COREF as acting contributively here, but making an underspecified contribution to its dialogue task. If the state is `s2209`, then COREF's contribution is $i_{3,2}$. If the state is `s2199`, then COREF's contribution is $i_{3,1}$. COREF accepts the utterance because it is willing to be recognized as making either contribution. In this way, COREF models itself as having extended two threads of coherent interpretation which it spawned upon perceiving the ambiguity in the user's utterance of *ok*.

3.3.9.1.3 Generating a contributive public motor action COREF assimilates the generation of non-verbal motor actions into the same general framework it uses for utterances. Because COREF's task models determine when non-verbal motor actions can occur coherently, COREF includes non-verbal motor actions in its horizon graph right alongside dialogue acts. Generating a contribution by way of a public motor action is thus a very similar process to generating a contributive utterance.

When COREF examines its acceptable horizon graph, it may occur that a public motor action is judged an acceptable contribution to the task. For example, Figure 3.32 shows COREF's acceptable horizon graph for dialogue state `s2593`, which is one of the viable states at time 5 in the subdialogue of Figure 3.3.

Figure 3.32: Acceptable horizon graph for COREF: `clickContinue`



Because the matcher, `s1`, has already added the object to their scene, COREF’s acceptability rules permit the agent to take a `continueTask[t3]` action here. For public motor actions like `continueTask`, COREF invokes its Sensory Event Interpreter to determine what interpretations the user might assign to the action. In this case, the Sensory Event Interpreter identifies the following interpretation:

$$i_{5,1} = i_{5,2} = \langle \text{COREF : tacitNop}[[s1 \text{ does say}[\text{yes}]]], \\ \text{COREF : continueTask}[\text{t3}] \rangle$$

This interpretation is supported in either of the two viable dialogue states, `s2593` and `s2583`, at time 5. Because COREF finds this contribution acceptable in either state, COREF’s Intention Generator accepts `continueTask[t3]` and returns action sequence $m^* = \langle \text{clickContinue}[] \rangle$, along with the Sensory Event Interpreter’s anticipated user interpretations. Because COREF expects to be recognized as having performed the same contribution regardless of which state is correct, we view COREF as acting contributively despite its uncertainty. Furthermore, COREF is thus able to complete this particular object successfully despite uncertainty about exactly which contributions have been made by the previous three utterances in the dialogue.

We will provide additional examples of COREF’s contributive language use in Chapter 4.

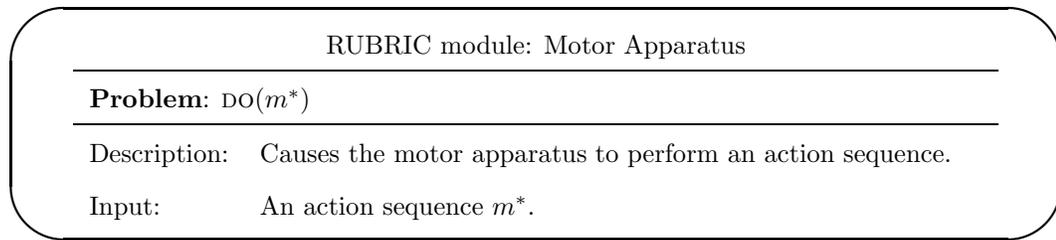


Figure 3.33: The Motor Apparatus

3.3.10 Motor Apparatus

The RUBRIC *Motor Apparatus* module is summarized in Figure 3.33. The RUBRIC architecture makes no assumptions about the types of motor actions that can be performed.

3.3.10.1 COREF's Motor Apparatus

COREF's complete set of possible motor actions, \mathcal{M} , is:

$$\mathcal{M} = \{ \text{privateClickToAdd}[\text{Object}], \\ \text{privateClickToRemove}[\text{Object}], \\ \text{clickContinue}[], \\ \text{clickSkip}[], \\ \text{say}[\text{S}] \\ \}$$

The clicking motor actions are self-explanatory. The motor action `say[S]` causes the agent to utter the string `S`. The effect is to make the string `S` appear as an utterance by the agent in the game interface.

This concludes our tour of the RUBRIC and COREF system architectures.

3.4 Conclusion

In this chapter, we have presented *contribution tracking* as a reasoning process by which uncertain agents can track the contributions that utterances make in dialogue. We have presented an agent

architecture, RUBRIC, which provides one approach to decomposing contribution tracking into sub-problems that can be solved by individual modules within an implemented agent. We have presented COREF as an example of an implemented RUBRIC agent that performs contribution tracking as it participates in an object identification game under uncertainty.

COREF's design emphasizes reversibility in several ways. COREF uses the same representations in interpretation as it uses in generation to reason about language, the horizon, and the individual contributions that interlocutors make. Further, it exploits the same update mechanisms to update its state representation after its own actions as it does after user actions. These features make it possible for COREF to understand, in principle, anything it can say itself. Additionally, these features make it possible for COREF to play either role — director or matcher — in its object identification game, as well as to play either role in the various subtasks that arise as part of the game.

In the next chapter, we use a user study to evaluate COREF's ability to perform contribution tracking in user-agent dialogues.

Chapter 4

Empirical results

In this chapter, we use a user study to evaluate COREF’s ability to perform contribution tracking in user-agent dialogues. We present our experimental method in Section 4.1. Section 4.2 presents an overview of the results of the user study, and in particular, relates COREF’s object outcomes to the uncertainty it faces while discussing objects with its users. In Section 4.3, we draw on this user study to articulate two new qualitative capabilities that COREF’s contribution tracking makes possible. We proceed in Section 4.4 to present our quantitative results. These include a quantitative analysis of COREF’s ability to reduce its own uncertainty, and a proof-of-concept demonstration that COREF’s experience in tracking its users’ contributions can be leveraged to improve its probabilistic interpretation model.

4.1 Method

We recruited 20 human subjects to carry out a series of collaborative reference tasks with COREF. Most of the subjects were undergraduate students participating for course credit at Rutgers University. The study was web-based; subjects participated from the location of their choice, and learned the task by reading on-screen instructions. The task instructions, which are included verbatim in Appendix A (page 210), informed each subject that they would work with an interactive dialogue agent rather than a human partner.

Each subject worked one-by-one through a series of 29 target objects, for a total of 580 objects and 3245 utterances across all subjects. For each subject, the 29 target objects were organized into 3 groups, with the first group of 4 in a 2x2 matrix, the next 9 in a 3x3 matrix, and the final 16 in a 4x4 matrix. As each object was completed, the correct target was removed from its group, leaving one fewer object in the matrix containing the remaining targets. The roles of director and matcher alternated with each group of objects. Either COREF or the subject was randomly chosen to be director first.

The experiment interface, which is depicted in Figures 3.1 (page 75) and 3.2 (page 76), allows an

<i>correct</i>	<i>no object</i>	<i>skipped</i>	<i>wrong</i>
75.0%	14.3%	7.4%	3.3%

Table 4.1: Overall distribution of object outcomes.

1 context	2 contexts	3 contexts
83.4%	6.8%	9.8%

Table 4.2: Number of possible contexts perceived when utterances or actions occur.

object to be completed with one of four outcomes. At any time, the matcher can click on an object to add it to their scene, which is another matrix containing previously added objects for the same group. An object is completed when the director presses either the `Continue (next object)` button or the `Skip this object` button, or when the matcher presses the `Skip this object` button. An outcome is scored *correct* if the director presses `Continue (next object)` after the matcher has added the correct target to her scene. It is scored *skipped* if the human subject presses the `Skip this object` button. (COREF never presses the `Skip this object` button.) It is scored *no object* or *wrong* if the director presses `Continue (next object)` before the matcher adds any object, or after the matcher adds the wrong object, respectively.

4.2 Overview of results

Table 4.1 summarizes COREF’s overall performance in the task. In this section, we provide an overview of how this performance relates to COREF’s uncertainty about the context. We will discuss more specific aspects of COREF’s uncertainty and contribution tracking in subsequent sections.

To begin, Table 4.2 shows the distribution in the number of contexts perceived as viable by COREF across all subjects and dialogue events. These data show that COREF is usually completely certain what the context is, but is uncertain 16.6% of the time.¹

To better understand how this uncertainty affects object outcomes, we investigated the agent’s performance during the subdialogues associated with individual objects. For example, Figure 3.3 (page 81 in Chapter 3) provides one example of a subdialogue associated with a single object which occurred in this user study. That particular subdialogue includes 4 utterances. The mean length of the 580 single object subdialogues was 5.0 utterances; Figure 4.1 shows the overall distribution in object subdialogue length.

¹Since COREF truncates its uncertainty at 3 possible contexts, the higher frequency of 3 possible contexts relative to 2 here masks a longer underlying tail.

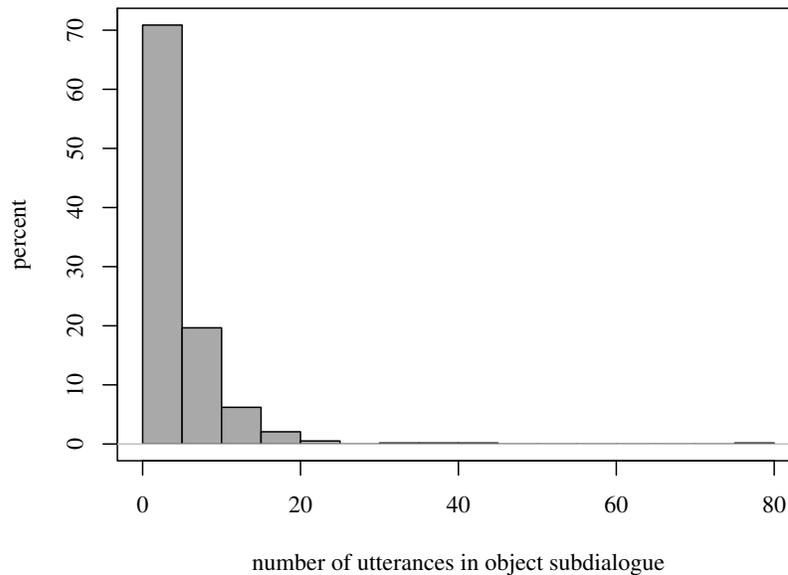


Figure 4.1: Length of object subdialogues.

For 436 (75.2%) of the 580 object subdialogues, COREF did not face any uncertainty about the context at any point. In the remaining 144 (24.8%) object subdialogues, COREF faced varying amounts of uncertainty. To summarize the effect of this uncertainty on the agent’s performance, we computed the mean uncertainty for each subdialogue. We defined the mean uncertainty for a subdialogue as the mean number of contexts that COREF viewed as viable when it interpreted the actions and utterances that occurred during the subdialogue (including actions and utterances performed both by the user and by COREF). For example, the mean uncertainty for the object subdialogue that spans times 1-5 in Figure 3.3 is 1.6 contexts. The mean of these mean uncertainties for all objects is 1.2 contexts; Figure 4.2 shows the overall distribution in mean uncertainty during the object subdialogues.

We will now begin to investigate the relationship between the agent’s uncertainty and the outcomes of its object subdialogues. Perhaps the simplest measure of COREF’s overall task performance is the frequency with which it achieves *correct* object outcomes as opposed to any of the “not correct” outcomes: *no object*, *skipped*, and *wrong object*. (We have seen in Table 4.1 that COREF achieves *correct* outcomes for 75.0% of the objects.) An introductory question, then, is whether there is any

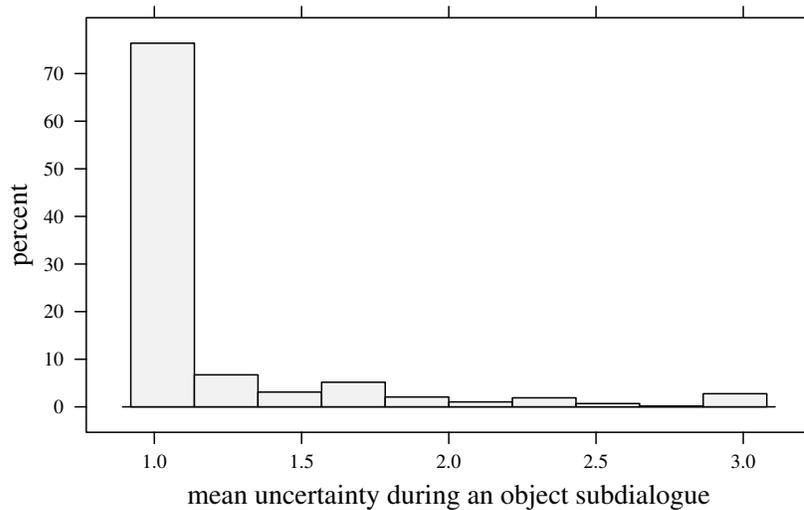


Figure 4.2: Mean uncertainty during object subdialogues.

relationship between COREF’s achieving a *correct* object outcome and the agent’s mean uncertainty during the object subdialogue.

Figure 4.3 shows the agent’s mean uncertainty during object subdialogues as a function of the outcomes of those subdialogues. For *correct* object subdialogues, the agent’s mean uncertainty is distributed with mean 1.12 contexts and variance 0.07, while for *not correct* object subdialogues, the agent’s mean uncertainty is distributed with mean 1.43 contexts and variance 0.49. These distributions are significantly different by a non-parametric Wilcoxon rank sum test ($W = 25533, p < 0.00001$).² This difference shows that task success, understood in terms of *correct* vs. *not correct* object outcomes, is indeed related (somehow) to the agent’s mean uncertainty during the object subdialogue.

Figure 4.4 looks more closely at how the agent’s mean uncertainty relates to the different kinds of *not correct* object outcomes that can occur. By pair-wise Wilcoxon rank sum tests, the only significant differences in these distributions are between *correct* and *no object* outcomes ($W = 14309.5, p < 0.0001$), and between *correct* and *skipped* outcomes ($W = 7594, p < 0.005$). These differences show that the agent’s success at achieving a *correct* outcome, rather than a *skipped* or *no object* outcome, is related (somehow) to the agent’s mean uncertainty during the object subdialogue.

²The distribution of the agent’s mean uncertainty is not normal. We therefore present our results in terms of the Wilcoxon rank sum test. For perspective, each Wilcoxon rank sum test reported in this section has also been compared to the results of a t-test, and the significance level always fell in the same general range.

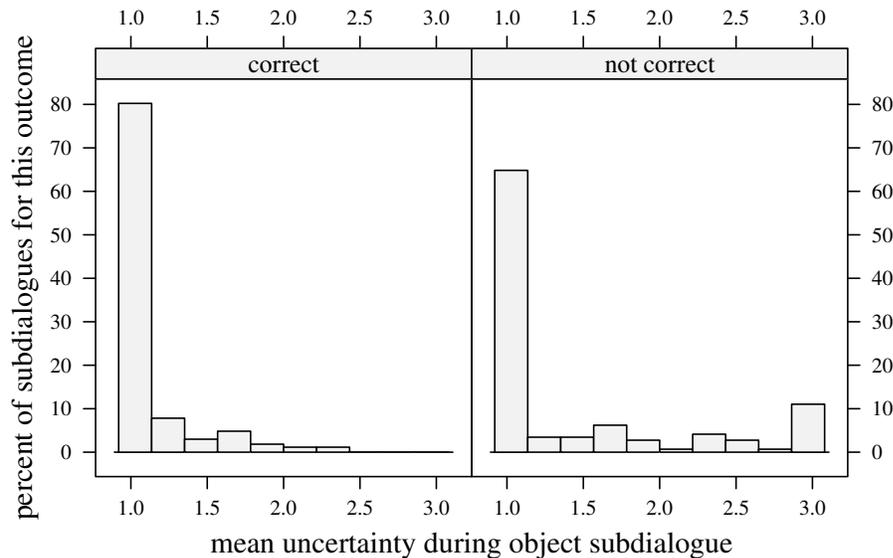


Figure 4.3: Mean uncertainty vs. *correct* object outcome during the object subdialogue. There are 435 objects (75.0%) in the *correct* bin, and 145 objects (25.0%) in the *not correct* bin.

However, these observed differences do not explain this relationship in causal terms. We will discuss in detail how the agent’s uncertainty can result in *no object* and *skipped* outcomes in Section 4.2.2.

For comparison, Figure 4.5 reverses our current perspective and shows object outcomes as a function of the agent’s mean uncertainty during object subdialogues. The figure shows that COREF’s performance, when measured by *correct* object outcomes, is somewhat robust to small-to-moderate amounts of uncertainty during object subdialogues. The two leftmost bins in the figure, where COREF faces a mean uncertainty of between 1 and 2.3 contexts during an object subdialogue, represent the vast majority (95.3%) of the object subdialogues, as can be seen in Figure 4.2. This essentially corresponds to the tracking of one or two threads of interpretation during a subdialogue. A small percentage of the time, however, COREF faces a higher mean uncertainty during an object subdialogue, between 2.3 and 3 contexts, and this higher mean uncertainty seems to have a large negative impact on object outcomes. We discuss this negative impact further in Section 4.2.2.

In total, 13.1% of COREF’s *correct* object outcomes occur at a moment when COREF is uncertain what the context is (9.7% occur when COREF views two contexts as viable, and 3.4% occur when COREF views three contexts as viable). These results show that maintaining a certain representation of a single thread of interpretation throughout a subdialogue is not strictly necessary for task success in COREF’s object identification game; on the other hand, uncertainty clearly does

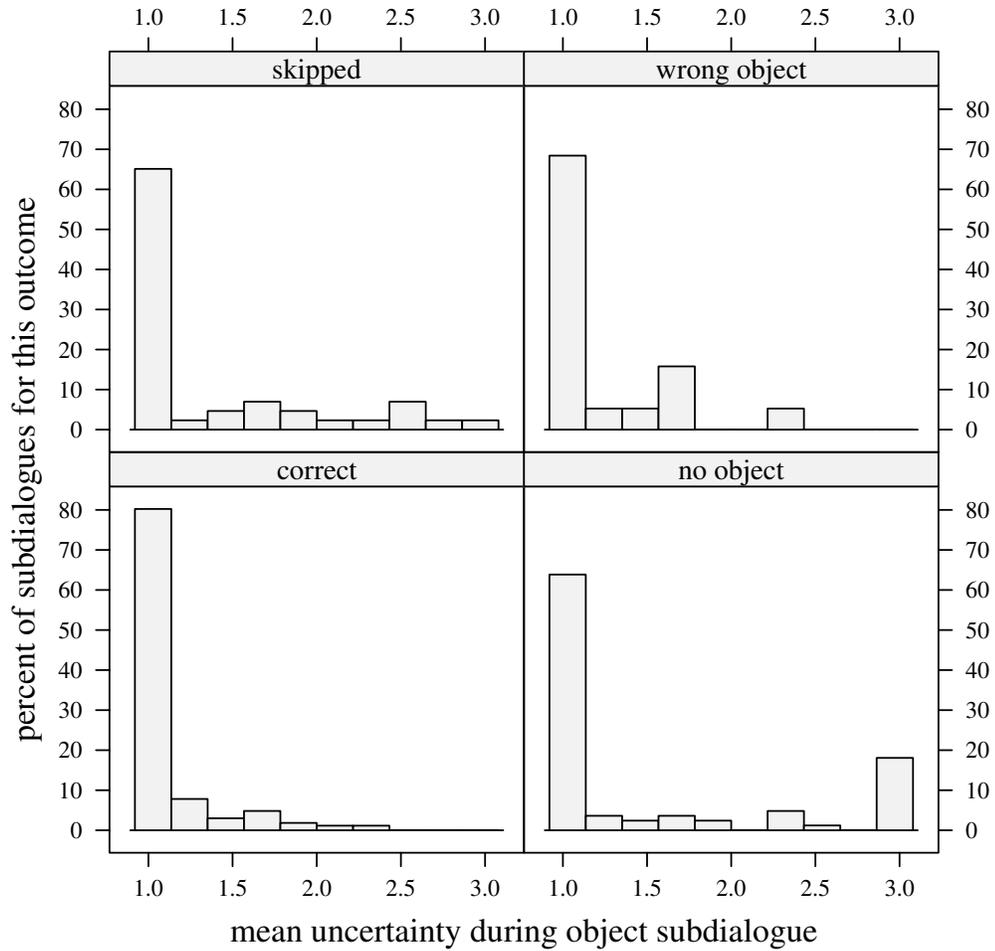


Figure 4.4: Mean uncertainty vs. object outcome during the object subdialogue. There are 435 objects (75.0%) in the *correct* bin, 83 objects (14.3%) in the *no object* bin, 43 objects (7.4%) in the *skipped* bin, and 19 objects (3.3%) in the *wrong object* bin.

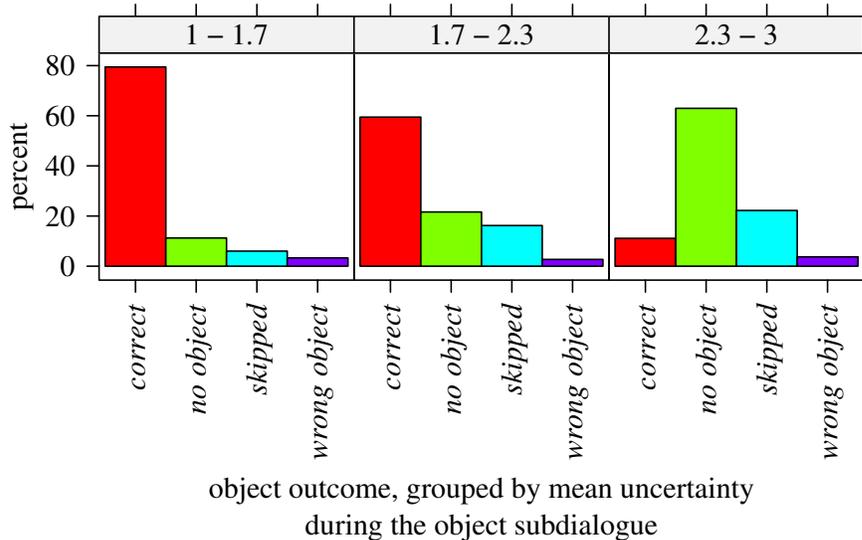


Figure 4.5: Object outcome vs. mean uncertainty during the object subdialogue. There are 516 objects (88.97%) in the leftmost bin, 37 objects (6.38%) in the middle bin, and 27 objects (4.66%) in the rightmost bin.

play a role in object outcome. To illustrate how these differing amounts of uncertainty relate to object outcome, we now give a few concrete examples.

4.2.1 Collaborative reference while certain about the context

As discussed above, by far (75.2%) the most common type of object subdialogue is one in which COREF does not face any uncertainty about the context. Figure 4.6 provides a simple example of this sort. In this example, COREF plays the role of director, and initiates collaborative reference to a new target object by contributively uttering *the dark orange circle*. COREF’s contextual interpretation algorithm allows it to view this utterance as unambiguously implicating that subject a20 and COREF are moving on to their next collaborative reference task, in which COREF will identify the new target, `t18`, to a20, and as further contributing that `t18` is equal to object `e1811228_4`. When the user does not immediately respond, COREF decides to ask *did you add it?* With this question, COREF implicates that the user has identified the object, and implicates that COREF has started a `Remind` task. The user responds *yes*, contributing that they have indeed added the object, so COREF decides to click the `Continue (next object)` button to move on to the next object. Because a20 has in fact added the right object to their scene, a *correct* outcome is achieved.

Figure 4.6: COREF achieves a *correct* outcome with a single thread of interpretation

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p> </div> <div style="text-align: center;"> <p>Your scene</p> </div> </div>	...	
Time:	1	2	
SE:	COREF: <i>the dark orange circle</i>	COREF: <i>did you add it ?</i>	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s53339</div> <div style="text-align: center;">$i_{1,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s53581</div> <div style="text-align: center;">$i_{2,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s53830</div> <div>...</div> </div> $i_{1,1} = \langle$ <ul style="list-style-type: none"> a20 : tacitNop[[COREF does clickContinue[]]], COREF : pushCollabRef[COREF, a20, t18], COREF : addcr[t18, equal(t18, e1811228_4)], COREF : setPrag[inFocus(Z), inFocus(t18)] \rangle $i_{2,1} = \langle$ <ul style="list-style-type: none"> a20 : tacitNop[[COREF does say[the dark orange circle]]], a20 : setVarValue[t18, e1811228_4], COREF : pushRemind[COREF, a20, past, refuseTaskAction, addToScene[e1811228_4]], COREF : askYNQ[past[a20, addToScene[e1811228_4]]], COREF : setPrag[inFocus(Y), inFocus(e1811228_4)] \rangle		
EI:	
Time:	3	4	
SE:	a20: <i>yes</i>	[COREF does clickContinue[]]	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s53830</div> <div style="text-align: center;">$i_{3,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s53923</div> <div style="text-align: center;">$i_{4,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s53994</div> <div>...</div> </div>		

continues on next page...

Figure 4.6: COREF achieves a *correct* outcome with a single thread of interpretation (continued)

```

i3,1= ⟨ a20 : tacitNop[[COREF does say[did you add it ?]]],
        a20 : addToScene[e1811228_4],
        a20 : past[a20, addToScene[e1811228_4]] ⟩
i4,1= ⟨ COREF : tacitNop[[a20 does say[yes]]],
        COREF : continueTask[t18] ⟩

```

Figure 4.7 provides another example of a *correct* object outcome for which COREF does not face any uncertainty during the object subdialogue. In this example, COREF plays the role of matcher, and begins by asking *is the target blue?* COREF views this utterance as unambiguously implicating that COREF and subject *c4* are moving on to the next collaborative reference task, in which COREF is asking whether the new target, *t6*, has the property which COREF represents as `darkblueFigureObject`. The user responds *yes*, contributing that it does indeed have that property, and follows this up by adding *blue square*. COREF interprets this as contributing that target *t6* is object *r6_1*, i.e. the blue square. Since COREF now knows that *r6_1* is the target object, COREF adds *r6_1* to its scene and says *click continue*, thereby implicating to the user that COREF has identified and added the object to its scene. When the user does not immediately respond, COREF says *done*, contributing that COREF previously added the object to its scene. The user responds by pressing **Continue (next object)** button, completing the object subdialogue.

Note that in interpreting the user's button press, COREF actually views the user's contribution as ambiguous. This happens because COREF's possible horizon graph for the user makes it coherent for the user to abandon the **Remind** task that COREF had previously pushed onto the task stack with its utterance *click continue*, and COREF is able to find coherent interpretations for the user's action either assuming that they are responding to COREF's reminder, or assuming that they are abandoning the reminder and deciding on independent grounds to press the continue button. However, because they are now moving onto the next object, COREF judges the ambiguity to be irrelevant and immediately drops one of these interpretations.

Figure 4.7: COREF achieves a *correct* outcome with a single thread of interpretation

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p> </div> <div style="text-align: center;"> <p>Your scene</p> </div> </div> <p>...</p>		
Time:	1	2	
SE:	COREF: <i>is the target blue ?</i>	c4: <i>yes</i>	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s5452</div> $i_{1,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s5622</div> $i_{2,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s7196</div> ... </div> <p>$i_{1,1} = \langle$ COREF : tacitNop[[c4 does clickContinue[]]], c4 : pushCollabRef[c4, COREF, t6], COREF : pushYNQ[COREF, c4, addcr[t6, P], negcr[t6, P]], COREF : askYNQ[addcr[t6, darkblueFigureObject(t6)]], COREF : setPrag[inFocus(Y), inFocus(t6)]\rangle</p> <p>$i_{2,1} = \langle$ c4 : tacitNop[[COREF does say[is the target blue ?]]], c4 : addcr[t6, darkblueFigureObject(t6)]\rangle</p>		
EI:	
Time:	3	4	
SE:	c4: <i>blue square</i>	COREF: <i>click continue</i>	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s7196</div> $i_{3,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s8470</div> $i_{4,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s8598</div> ... </div> <p>$i_{3,1} = \langle$ COREF : tacitNop[[c4 does say[yes]]], c4 : addcr[t6, equal(t6, r6_1)], c4 : setPrag[inFocus(Z), inFocus(t6)]\rangle</p>		

continues on next page...

Figure 4.7: COREF achieves a *correct* outcome with a single thread of interpretation (continued)

```

i4,1 = ⟨ COREF : tacitNop[[c4 does say[blue square]]],
          COREF : setVarValue[t6, r6_1],
          COREF : addToScene[r6_1],
          COREF : pushRemind[COREF, c4, past, refuseTaskAction, continue-
Task[t6]],
          COREF : command[c4, continueTask[t6]]⟩
    
```

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p> </div> <div style="text-align: center;"> <p>Your scene</p> </div> </div> <p style="text-align: center;">...</p>		
Time:	5	6	
SE:	[perceived 9 figure objects]	COREF: <i>done</i>	
CT:	<pre> i_{5,1} = ⟨ COREF : perceive[PerceivedFigureObjectsEvent<[r8_1, r6_1, r13_1, e14_...]]⟩ i_{6,1} = ⟨ c4 : tacitNop[[COREF does say[click continue]]], COREF : past[COREF, addToScene[r6_1]]⟩ </pre>		

EI:	...		
Time:	7		
SE:	[c4 does clickContinue[]]		
CT:			

continues on next page...

Figure 4.7: COREF achieves a *correct* outcome with a single thread of interpretation (continued)

```

i7,1 = ⟨ c4 : tacitNop[[COREF does say[done]]],
           c4 : continueTask[t6]⟩
i7,2 = ⟨ c4 : tacitAbandonTasks[2],
           c4 : continueTask[t6]⟩

```

4.2.2 Collaborative reference under uncertainty about the context

As discussed above, 75.2% of COREF’s object subdialogues are completed under a single thread of interpretation. In the remaining 24.8%, COREF spawns multiple threads of interpretation at some point, and tries to proceed under some degree of uncertainty using contribution tracking. In this section, we provide several examples of COREF’s reasoning as it pursues more than one thread of interpretation during an object subdialogue, and illustrate the different outcomes that can occur.

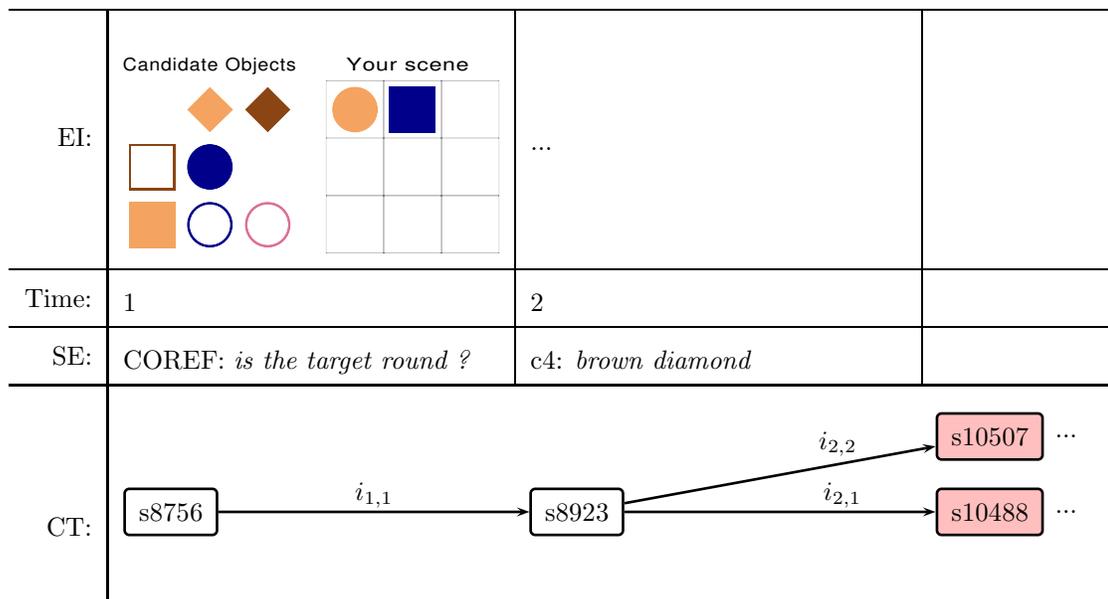
We have already discussed in detail one example of a *correct* object outcome that involved two threads of interpretation: that of Figure 3.3 (page 81) in Chapter 3. In that example, COREF spawned two threads of interpretation to capture a perceived ambiguity in a user’s utterance of *ok*. Figure 4.8 shows another example of a *correct* outcome following an ambiguous utterance, but this time the ambiguity arises in the interpretation of color vocabulary. In this example, COREF plays the role of matcher, and begins by asking *is the target round?* Subject *c4* responds *brown diamond*. COREF is able to interpret this utterance as coherent under the assumption that *c4* has abandoned COREF’s question, but perceives an ambiguity in the interpretation of the word *brown*: does *c4* mean for the semantic variable **Brown** (associated with the word *brown*) to be identified with the property COREF represents as `saddlebrownFigureObject`, or with the property COREF represents as `sandybrownFigureObject`?³ COREF thus spawns two threads of interpretation, one in which *c4* intended each of these meanings. COREF then attempts to clarify the user’s meaning by asking *do you mean dark brown?* In generating this clarification question, COREF’s perspective is that it isn’t sure which context it is in, but in either case, COREF anticipates that it will be recognized as contributing $i_{3,1} = i_{3,2}$, in which COREF pushes a clarification question about semantic variable **Brown** and performs the following dialogue act:

³Compare the differing shades of the light and dark brown diamonds in the figure.

askYNQ[addcr[Brown, equal(Brown, saddlebrownFigureObject)]]].

The user responds *yes*, which COREF interprets as contributing that the correct assignment of variable **Brown** is `saddlebrownFigureObject`. Importantly, COREF views this interpretation as only coherent in state `s10747`, in which `c4` previously meant `saddlebrownFigureObject` when they said *brown*. This allows COREF to eliminate one of the threads of interpretation it had spawned; the elimination of a thread of interpretation is indicated in the figure with a red 'X'. With its uncertainty resolved, COREF proceeds by simply saying *done*, which it views as unambiguously contributing that COREF has identified the value of the variable **Brown** (by implication), that COREF has identified the target object (by implication), that COREF has added the target object to its scene (by implication), and that its addition of the target object occurred in the past (a dialogue act). The dialogue then proceeds to a *correct* outcome. We discuss COREF's clarification strategy further, including its relation to alternative approaches, in Section 4.3.2 (page 174).

Figure 4.8: COREF achieves a *correct* outcome after clarifying the user's meaning



continues on next page...

Figure 4.8: COREF achieves a *correct* outcome after clarifying the user's meaning (continued)

```

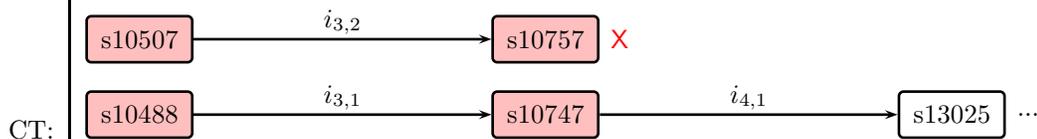
i1,1 = ⟨ COREF : tacitNop[[c4 does clickContinue[]]],
          c4 : pushCollabRef[c4, COREF, t7],
          COREF : pushYNQ[COREF, c4, addcr[t7, P], negcr[t7, P]],
          COREF : askYNQ[addcr[t7, circleFigureObject(t7)]],
          COREF : setPrag[inFocus(Y), inFocus(t7)]⟩

i2,1 = ⟨ c4 : tacitAbandonTasks[2],
          c4 : addcr[t7, rhombusFigureObject(t7)],
          c4 : setPrag[inFocus(Y), inFocus(t7)],
          c4 : addcr[t7, saddlebrownFigureObject(t7)]⟩

i2,2 = ⟨ c4 : tacitAbandonTasks[2],
          c4 : addcr[t7, rhombusFigureObject(t7)],
          c4 : setPrag[inFocus(Y), inFocus(t7)],
          c4 : addcr[t7, sandybrownFigureObject(t7)]⟩

```

EI:	
Time:	3	4	
SE:	COREF: <i>do you mean dark brown ?</i>	c4: <i>yes</i>	



```

i3,1 = ⟨ COREF : pushClarify[[c4 does say[brown diamond]], c4, COREF,
          Brown],
          COREF : pushYNQ[COREF, c4, addcr[Brown, P], negcr[Brown, P]],
          COREF : askYNQ[addcr[Brown, equal(Brown, saddlebrownFigureObject)]]⟩

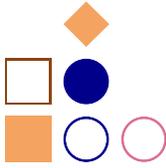
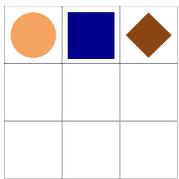
```

continues on next page...

Figure 4.8: COREF achieves a *correct* outcome after clarifying the user's meaning (continued)

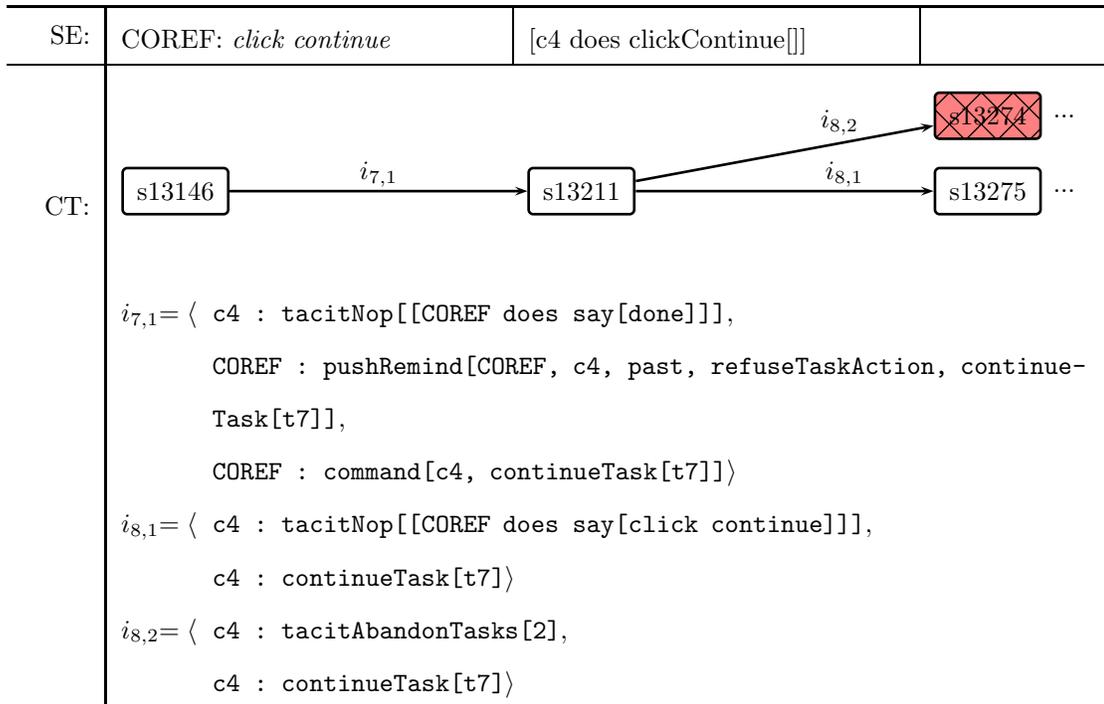
```

i3,2 = ⟨ COREF : pushClarify[[c4 does say[brown diamond]], c4, COREF,
          Brown],
          COREF : pushYNQ[COREF, c4, addcr[Brown, P], negcr[Brown, P]],
          COREF : askYNQ[addcr[Brown, equal(Brown, saddlebrownFigureOb-
          ject)]]⟩
i4,1 = ⟨ c4 : tacitNop[[COREF does say[do you mean dark brown ?]],
          c4 : addcr[Brown, equal(Brown, saddlebrownFigureObject)]]⟩
    
```

EI:	...	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div>	
Time:	5	6	
SE:	COREF: <i>done</i>	[perceived 9 figure objects]	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s13025</div> <div style="text-align: center;">→^{<i>i</i>_{5,1}}</div> <div style="border: 1px solid black; padding: 2px 10px;">s13141</div> <div style="text-align: center;">→^{<i>i</i>_{6,1}}</div> <div style="border: 1px solid black; padding: 2px 10px;">s13146</div> <div style="text-align: center;">...</div> </div> <p><i>i</i>_{5,1} = ⟨ COREF : tacitNop[[c4 does say[yes]]], COREF : setVarValue[Brown, saddlebrownFigureObject], COREF : setVarValue[t7, rh7_2], COREF : addToScene[rh7_2], COREF : past[COREF, addToScene[rh7_2]]⟩</p> <p><i>i</i>_{6,1} = ⟨ COREF : perceive[PerceivedFigureObjectsEvent<[r8_2, r13_2, e14_2, e10...]]⟩</p>		
EI:	
Time:	7	8	

continues on next page...

Figure 4.8: COREF achieves a *correct* outcome after clarifying the user's meaning (continued)



When COREF encounters uncertainty in interpreting a user utterance, COREF is not always able to resolve the competing threads of interpretation as cleanly as it does in Figure 4.8. While the resolution of the agent's uncertainty is not necessary for a *correct* outcome (as the example of Figure 3.3 shows), the distribution of outcomes shown in Figure 4.5 (page 148) indicates that higher uncertainty during an object subdialogue is associated with a higher frequency of *no object* and *skipped* outcomes. These outcomes usually occurred when the human subject was playing the role of director and clicked the **Continue (next object)** button before COREF added any object (yielding a *no object* outcome), or when the human subject was playing either role and clicked the **Skip this object** button. Both these outcomes tended to occur when the subject seemed to become frustrated with a lack of progress for an object. In particular, rather than following the instructions to the letter by using the **Skip this object** button whenever they felt they were no longer making progress, we observed that when our subjects played the role of director, if they became frustrated, they often clicked these two buttons interchangeably in order to move on to the next object. The result is that both *no object* and *skipped* outcomes tend to indicate subject frustration in this data set.

In Figure 4.9, we provide an example in which COREF faces relatively high mean uncertainty

and a *skipped* outcome occurs. In this object subdialogue, subject a22 plays the role of director, and initiates collaborative reference to a new object by saying *the solid blue square*. COREF thinks there are two solid blue squares which the user might mean, viz. r2697701_1 and r2697724_1, and so spawns two threads of interpretation. In this example, before COREF can respond (it would probably respond by asking a clarification question), subject a22 continues by asking *did you add it?* Here, the human subject mirrors COREF's common usage of *did you add it?* as a follow-up question when COREF is playing the role of director. Due to COREF's reversible linguistic reasoning, COREF is able to interpret a22's question as a reminder to add the object. However, COREF isn't sure which object is under discussion, and so COREF extends its two threads of interpretation from time 2 to time 3.

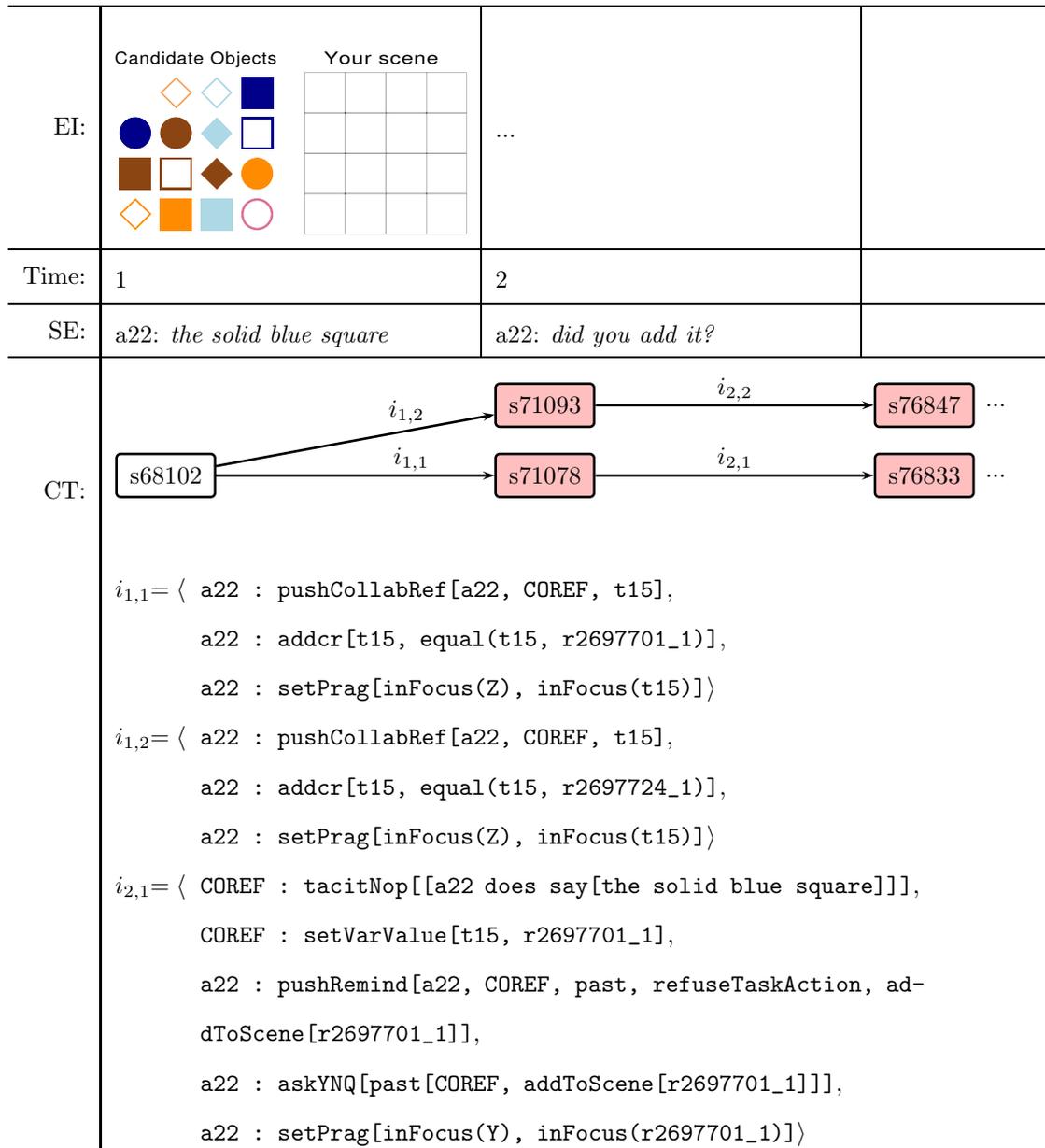
Ideally, at this point COREF would issue some follow-up question which would allow it to resolve the ambiguity and achieve a *correct* outcome. Unfortunately for COREF, it chooses to utter *do you mean it?* at time 3. COREF's reasoning about this utterance highlights one of the dangers of underspecifying your dialogue contributions in the way that COREF does. What has gone wrong, in particular, is that COREF has anticipated that its use of *it* would be recognized as meaning r2697701_1 if the context were s76833, and as meaning r2697724_1 if the context were s76847. The reason is that the semantic constraint that COREF's grammar associates with *it* requires that the referent of *it* be in focus, and COREF's contextual model tells it that different objects are in focus in these two contexts. Further, COREF's policy for accepting underspecified intentions admits this utterance, on the grounds that COREF is willing to perform either of these two dialogue acts in the corresponding context:

askYNQ[addcr[X62, equals(X62, r2697701_1)]] (in context s76833)
 askYNQ[addcr[X62, equals(X62, r2697724_1)]] (in context s76847)

Our perspective is that making such an underspecified contribution is not generally a problem in itself; however, it is self-defeating to perform a clarification question that underspecifies exactly the information you are seeking to clarify! This is demonstrated by the response of COREF's interlocutor (who is perhaps puzzled by COREF's question), which is simply *yes*. COREF interprets the subject's answer of *yes* in the usual way, which extends its two threads of interpretation: if the context was s77081, then X62 (the user's original intended referent) is r2697701_1; if the context was s77091, then X62 is r2697724_1. COREF is left in exactly the same epistemic situation it was in before it asked *do you mean it?*, and therefore must proceed with its original two threads of interpretation

unresolved.

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation



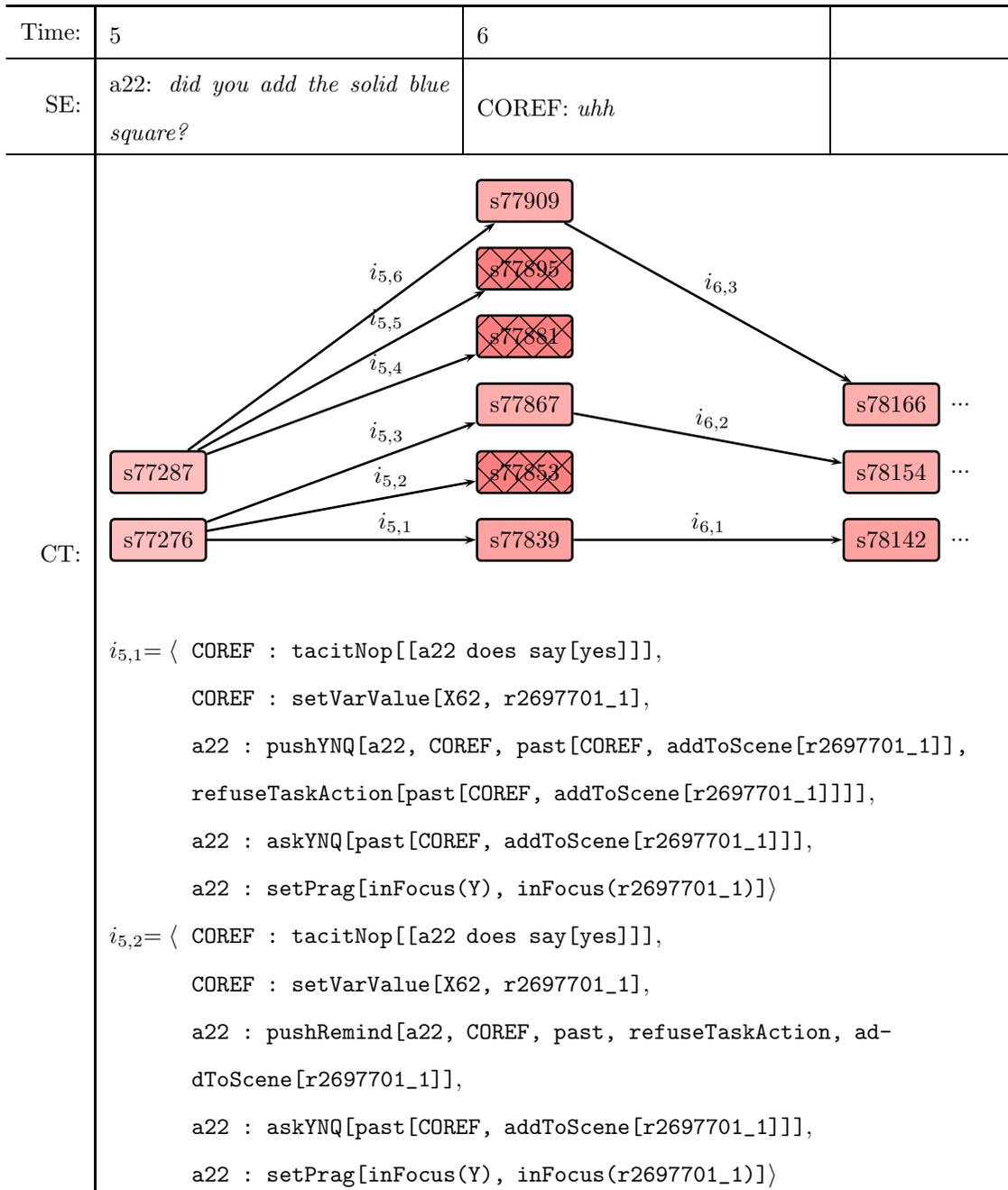
continues on next page...

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation (continued)

		$i_{2,2} = \langle$ COREF : tacitNop[[a22 does say[the solid blue square]]], COREF : setVarValue[t15, r2697724_1], a22 : pushRemind[a22, COREF, past, refuseTaskAction, addToScene[r2697724_1]], a22 : askYNQ[past[COREF, addToScene[r2697724_1]]], a22 : setPrag[inFocus(Y), inFocus(r2697724_1)] \rangle	
EI:	
Time:	3	4	
SE:	COREF: <i>do you mean it ?</i>	a22: <i>yes</i>	
CT:			
	$i_{3,1} = \langle$ COREF : pushClarify[[a22 does say[<i>did you add it?</i>]], a22, COREF, X62], COREF : pushYNQ[COREF, a22, addcr[X62, P], negcr[X62, P]], COREF : askYNQ[addcr[X62, equal(X62, r2697701_1)]] \rangle		
	$i_{3,2} = \langle$ COREF : pushClarify[[a22 does say[<i>did you add it?</i>]], a22, COREF, X62], COREF : pushYNQ[COREF, a22, addcr[X62, P], negcr[X62, P]], COREF : askYNQ[addcr[X62, equal(X62, r2697724_1)]] \rangle		
	$i_{4,1} = \langle$ a22 : tacitNop[[COREF does say[<i>do you mean it ?</i>]]], a22 : addcr[X62, equal(X62, r2697701_1)] \rangle		
	$i_{4,2} = \langle$ a22 : tacitNop[[COREF does say[<i>do you mean it ?</i>]]], a22 : addcr[X62, equal(X62, r2697724_1)] \rangle		
EI:	

continues on next page...

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation (continued)



continues on next page...

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation (continued)

```

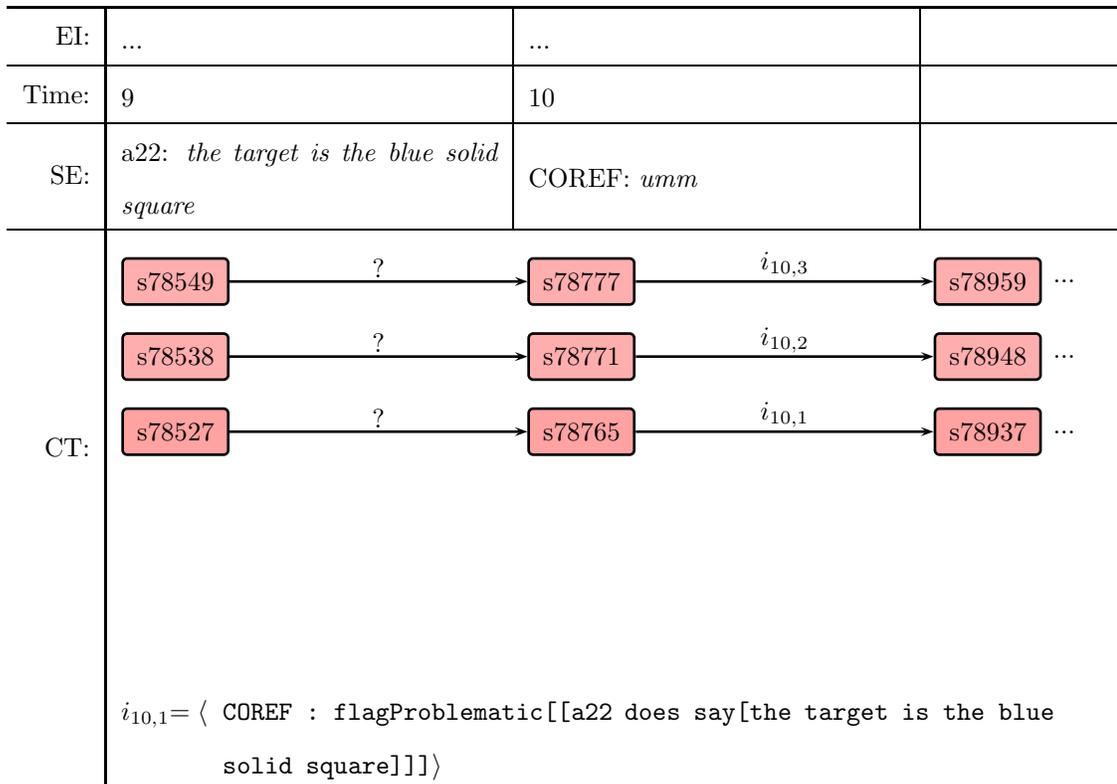
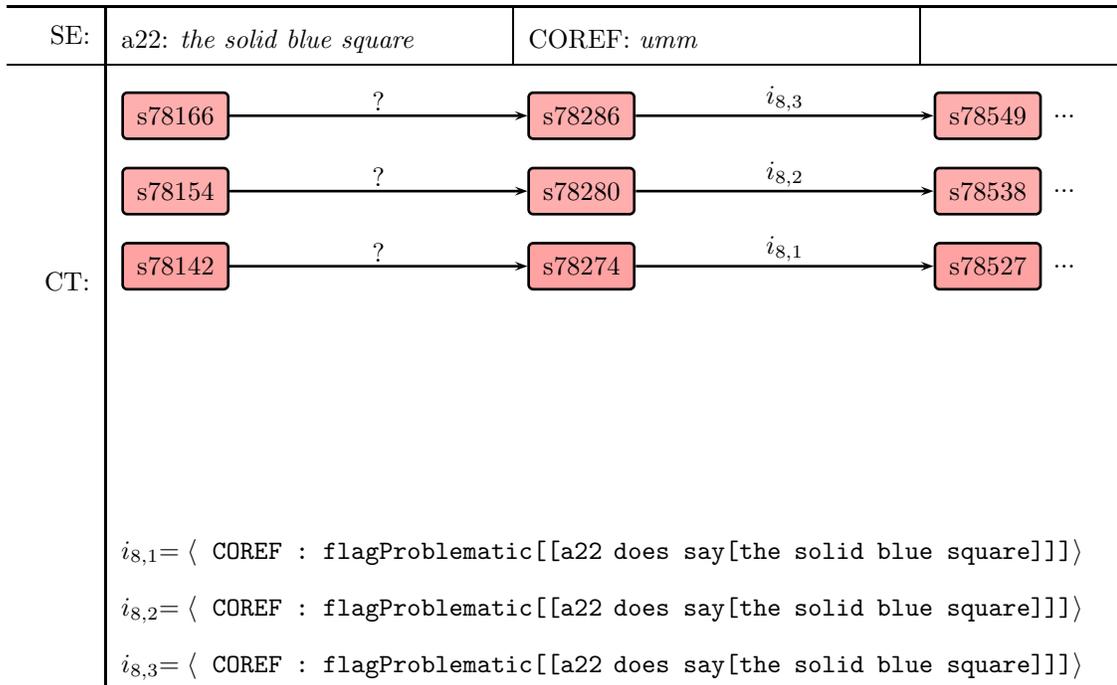
i5,3= ⟨ a22 : tacitAbandonTasks[4],
        a22 : pushRemind[a22, COREF, past, refuseTaskAction, ad-
        dToScene[r2697701_1]],
        a22 : askYNQ[past[COREF, addToScene[r2697701_1]]],
        a22 : setPrag[inFocus(Y), inFocus(r2697701_1)]⟩
i5,4= ⟨ COREF : tacitNop[[a22 does say[yes]]],
        COREF : setVarValue[X62, r2697724_1],
        a22 : pushYNQ[a22, COREF, past[COREF, addToScene[r2697724_1]],
        refuseTaskAction[past[COREF, addToScene[r2697724_1]]]],
        a22 : askYNQ[past[COREF, addToScene[r2697724_1]]],
        a22 : setPrag[inFocus(Y), inFocus(r2697724_1)]⟩
i5,5= ⟨ COREF : tacitNop[[a22 does say[yes]]],
        COREF : setVarValue[X62, r2697724_1],
        a22 : pushRemind[a22, COREF, past, refuseTaskAction, ad-
        dToScene[r2697724_1]],
        a22 : askYNQ[past[COREF, addToScene[r2697724_1]]],
        a22 : setPrag[inFocus(Y), inFocus(r2697724_1)]⟩
i5,6= ⟨ a22 : tacitAbandonTasks[4],
        a22 : pushRemind[a22, COREF, past, refuseTaskAction, ad-
        dToScene[r2697724_1]],
        a22 : askYNQ[past[COREF, addToScene[r2697724_1]]],
        a22 : setPrag[inFocus(Y), inFocus(r2697724_1)]⟩
i6,1= ⟨ COREF : abandonTasks[3]⟩
i6,2= ⟨ COREF : abandonTasks[2]⟩
i6,3= ⟨ COREF : abandonTasks[2]⟩

```

EI:	
Time:	7	8	

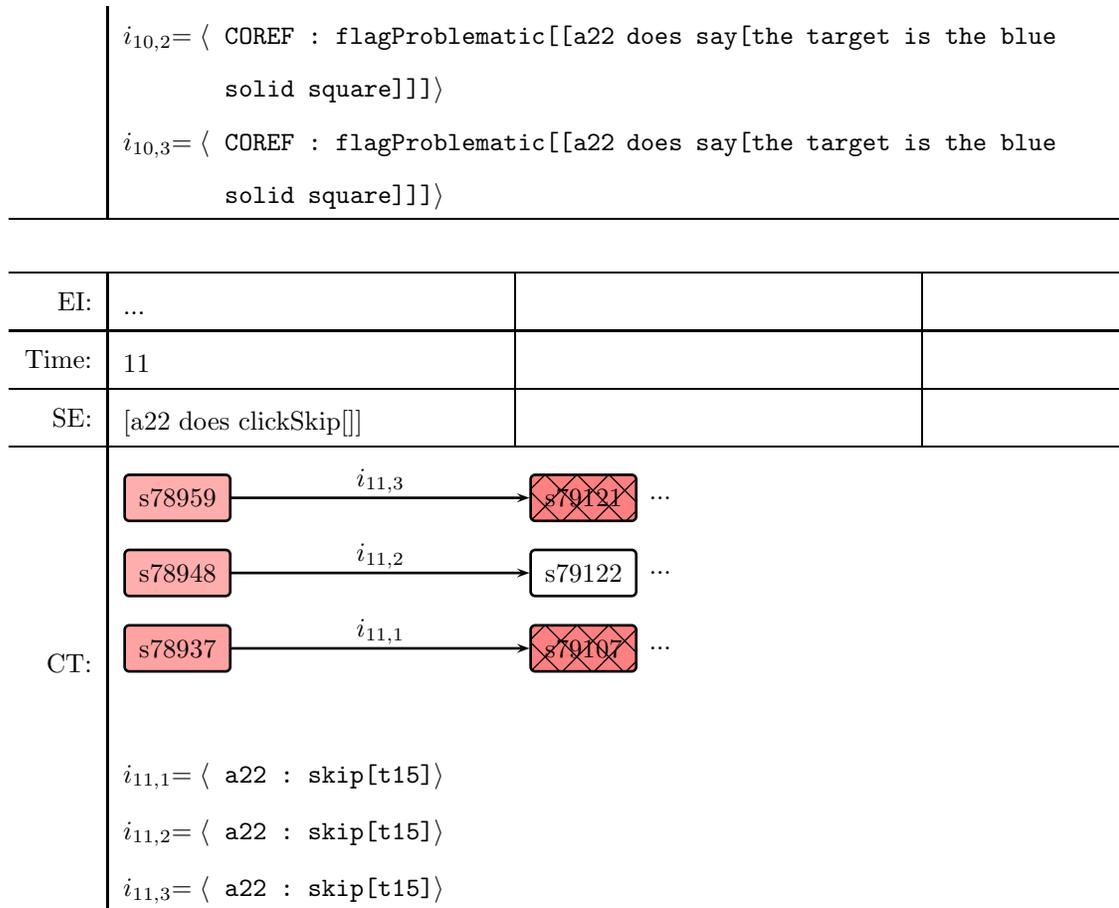
continues on next page...

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation (continued)



continues on next page...

Figure 4.9: COREF achieves a *skipped* outcome while pursuing several threads of interpretation (continued)



Subject a22 proceeds by asking *did you add the solid blue square?* As could be expected, COREF perceives this new utterance as ambiguous due to the ambiguity of *the solid blue square*. Worse, this ambiguity multiplies the agent's previous uncertainty about which context it is in. Further, COREF perceives additional ambiguity in the implicatures associated with this utterance. After this utterance, at time 6, COREF is sufficiently confused that it chooses to contribute that it has abandoned the user's question, which it does by saying *uhh*. Subject a22 responds by repeating its (problematic) referring expression *the solid blue square*. COREF finds no way of interpreting this repeated reference as a coherent contribution, and so in response COREF contributes that it had trouble interpreting *the solid blue square* by saying *umm* at time 8. The user tries once more to identify the object to COREF, but eventually decides just to skip the object at time 11. A *skipped* outcome occurs.

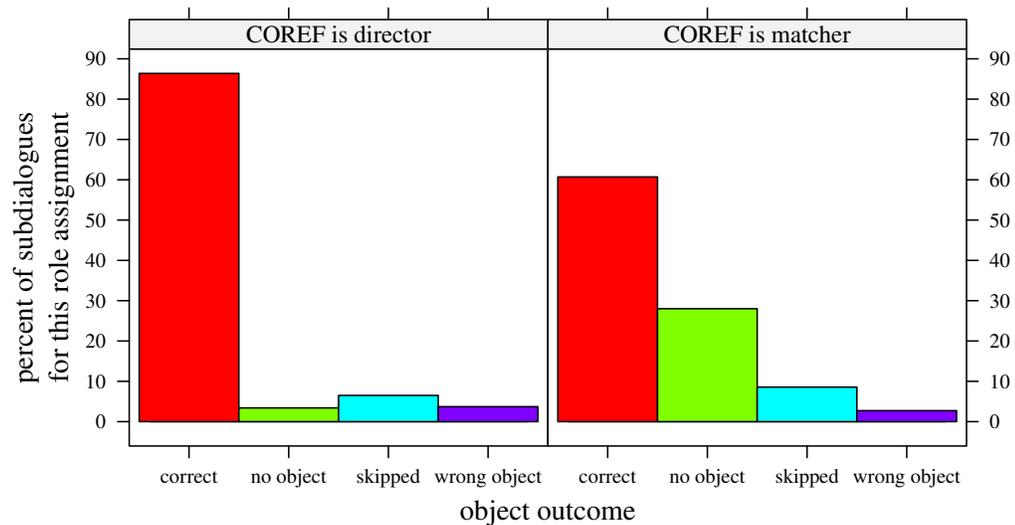


Figure 4.10: Object outcome vs. role assignment. There are 323 objects (55.7%) in the *COREF is director* bin, and 257 objects (44.3%) in the *COREF is matcher* bin.

This object subdialogue, in which COREF’s mean uncertainty is 2.5 contexts, provides a good model of what can happen when COREF faces a problematic ambiguity which it is unable to resolve. Essentially, COREF can become confused and find itself with no foreseeable way to make a coherent contribution that moves the dialogue forward. In this case, the user may well become frustrated and click a button to move on to the next object, yielding a *skipped* or *no object* outcome. Such dialogues exhibit some of the limitations in COREF’s competence as a dialogue partner. However, it should perhaps be emphasized that, as Figures 4.2 (page 145) and 4.5 (page 148) make clear, these problematic object subdialogues are a small minority of the object subdialogues that occurred in this user study.

4.2.3 Effect of agent’s role on task performance

COREF has been designed to be able to play either role, director or matcher, in its object identification games. In this section, we discuss the agent’s observed performance in these two roles.

Figure 4.10 shows the object outcomes that COREF achieved in each role. As the figure shows, the agent’s role assignment for an object has a significant effect on object outcome ($\chi^2(3, N = 580) = 74.4, p < 0.0001$). In particular, COREF achieves *correct* object outcomes more frequently when it is director than when it is matcher. There seem to be two main contributing factors here. The

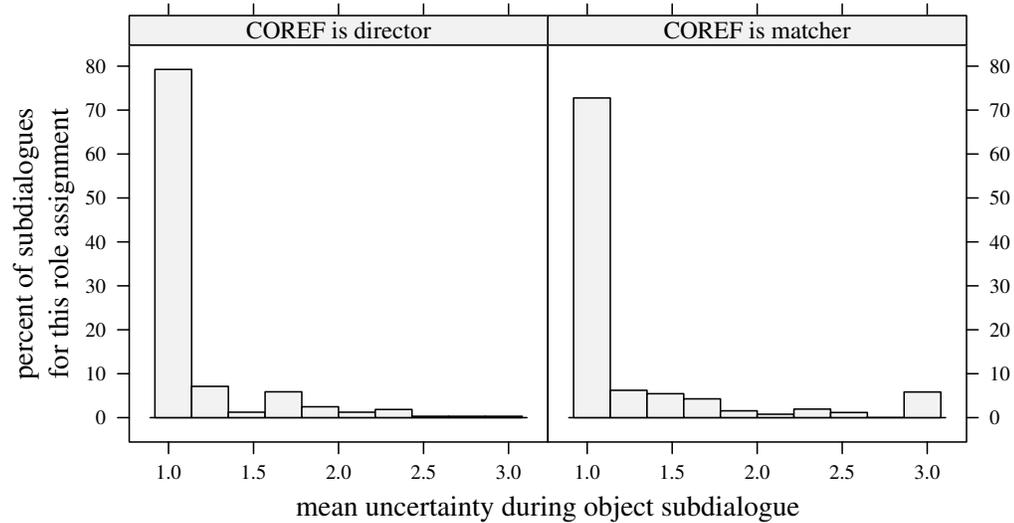


Figure 4.11: Mean uncertainty vs. role assignment. There are 323 objects (55.7%) in the *COREF is director* bin, and 257 objects (44.3%) in the *COREF is matcher* bin.

first is that, when COREF is matcher, it is likely to encounter user-generated object descriptions which lie outside its grammar and are difficult for the agent to interpret. This second is that, partly in response to this issue having arisen in development, COREF uses an aggressive question-asking policy when it is matcher. If the user does not immediately begin to describe the target object, this policy leads COREF to take the initiative and start asking yes-no questions about the target object. This can lead to a long sequence of yes-no questions that is not particularly efficient (e.g. *is the target round?* no. *is it brown?* yes. *is it solid?* no...). The incessant stream of yes-no questions sometimes seemed to frustrate the user enough that they would click to continue on to the next object before COREF had identified the target object (thereby achieving a *no object* rather than a *correct object* outcome).

Finally, Figure 4.11 shows the relationship between the agent’s uncertainty and its role assignment. The agent’s mean uncertainty as director was distributed with mean 1.14 contexts and variance 0.11, while its mean uncertainty as matcher was distributed with mean 1.26 contexts and variance 0.29. These distributions are significantly different by a non-parametric Wilcoxon rank sum test ($W = 38020.5, p < 0.05$). The slightly greater degree of uncertainty the agent faces when it is matcher seems to be due to uncertainty arising in interpreting user-generated object descriptions that lie partially outside its grammar.

4.3 New qualitative capabilities

In this section, we present two new qualitative capabilities that we have achieved in our implementation of contribution tracking in COREF.

4.3.1 Capturing the ambiguity of acknowledgments

It can be tempting to think of acknowledgments as a set of expression *types*, such as *okay*, *yes/yeah*, *right*, *m-hm*, *uh huh*, and others, which are distinguished from other types of expressions by the conventionalized role they play in allowing interlocutors to signal understanding of a previous or ongoing utterance (Clark and Schaefer, 1989). However, it is easy to see that speakers employ many of these same expression types for other uses as well. The word *yeah*, for example, can obviously be used to answer questions as well as to acknowledge utterances. Similarly, the word *right* can be used in several ways other than to signal understanding.⁴ Even the word *okay*, which may seem to be a paradigm candidate for an expression type whose sole or primary use is to acknowledge other utterances, in fact serves many different functions in real-world dialogue (Clark and Schaefer, 1989; Novick and Sutton, 1994; Core and Allen, 1997). In short, interlocutors often perform acknowledgments using general-purpose linguistic expressions, and they seem to rely on context to help make their intention to perform an acknowledgment using one of these expressions clear on a particular occasion.

The fact that acknowledgments are only interpretable as such in particular contexts might not seem to be any cause for alarm: just as with other kinds of utterances, interlocutors exploit context in order to generate a recognizable acknowledgment. However, as it turns out, interlocutors often do not make their intentions fully recognizable when they utter the expression types that are traditionally categorized as acknowledgments. For example, in an annotation reliability study using dialogues from the TRAINS corpus, Core and Allen (1997) found that annotators often had a hard time agreeing on whether an utterance was an acknowledgment or an acceptance. They give the following example:

s: so we'll take the train through Corning
u: okay
s: and on to Elmira.

In this example, it is hard to definitively judge whether *u* is simply acknowledging that they have understood *s*'s proposal (to take the train through Corning), or whether *u* is (also) signaling

⁴This is true even when *right* is used as a one word utterance. For example, *A:* which way should I go? *B:* right.

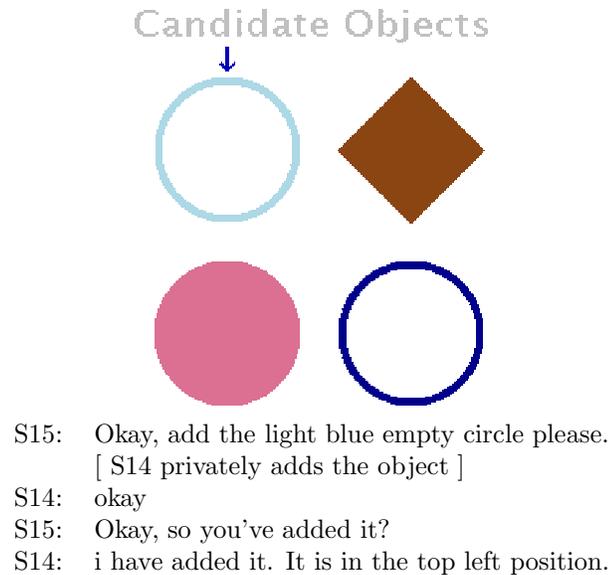


Figure 4.12: An ambiguous acknowledgment by subject S14 in a human-human dialogue.

acceptance of the proposal.

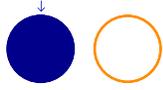
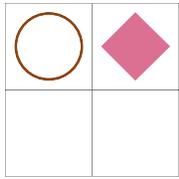
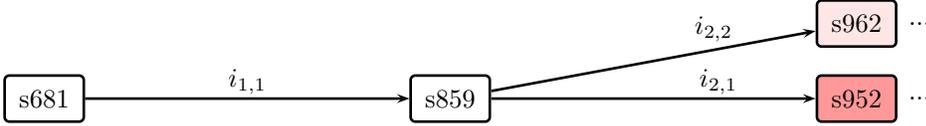
This kind of “ambiguous acknowledgment” is not unique to the TRAINS corpus. For example, Figure 4.12 provides an ambiguous use of *okay* which we observed in a naturally occurring fragment of human–human dialogue in COREF’s domain. In this interaction, two human subjects, S15 and S14, perform COREF’s object identification game together via teletype from separate rooms. S15 begins by instructing S14 to click on a certain object in S14’s display. S14 does so, but S15 cannot observe the action. This leads S15 to perceive an ambiguity when S14 says *okay*: has S14 merely acknowledged S15’s instruction, or has S14 also clicked the object? The ambiguity matters for this task, so S15 engages the ambiguity with a followup question.

Using contribution tracking, COREF is able to capture these kinds of ambiguous acknowledgments under alternative threads of interpretation. Figure 3.3 (page 81 in Chapter 3) provides one such example. COREF’s reasoning in that example provides one possible model of the kind of reasoning that seems to be occurring in the human-human dialogue in Figure 4.12.

Because COREF is designed to collaborate under uncertainty about what the previous contributions have been, and because acknowledgments like *okay* are modeled as just another kind of contribution in COREF’s task models, COREF is relatively flexible about how the ambiguity introduced by an ambiguous acknowledgment is resolved. For example, a user could respond to this type of *did you add it?* by saying *no* as in Figure 4.13. In this dialogue, which shows COREF’s

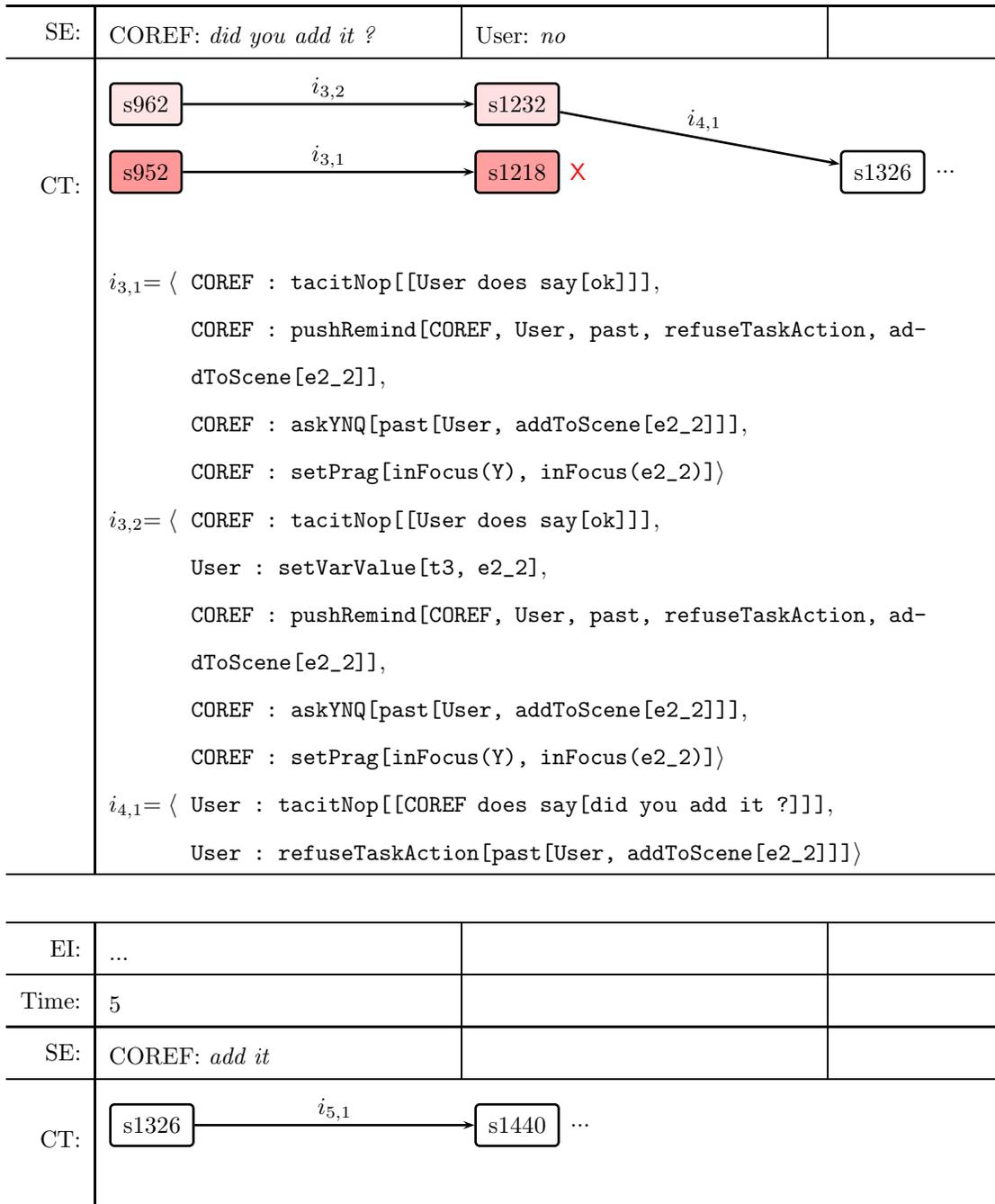
implemented contribution tracking during a hypothetical variation on the observed dialogue in Figure 3.3, the user’s response of *no* allows COREF to defuse its uncertainty and definitively interpret the user’s previous utterance of *ok* as a simple acknowledgment (modeled as a *nop* dialogue act, as discussed in Chapter 3). With its uncertainty resolved, COREF simply proceeds to issue a reminder to add the object.

Figure 4.13: COREF disambiguates an ambiguous acknowledgment

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div> <p style="text-align: center;">...</p>		
Time:	1	2	
SE:	COREF: <i>the blue circle</i>	User: <i>ok</i>	
CT:	 <pre> graph LR s681[s681] -- i1,1 --> s859[s859] s859 -- i2,2 --> s962[s962] s859 -- i2,1 --> s952[s952] </pre> <p> $i_{1,1} = \langle$ User : tacitNop[[COREF does clickContinue[]]], COREF : pushCollabRef[COREF, User, t3], COREF : addcr[t3, equal(t3, e2_2)], COREF : setPrag[inFocus(Z), inFocus(t3)]\rangle </p> <p> $i_{2,1} = \langle$ User : tacitNop[[COREF does say[the blue circle]]], User : setVarValue[t3, e2_2], User : addToScene[e2_2], User : past[User, addToScene[e2_2]]\rangle </p> <p> $i_{2,2} = \langle$ User : nop[[COREF does say[the blue circle]]]\rangle </p>		
EI:	
Time:	3	4	

continues on next page...

Figure 4.13: COREF disambiguates an ambiguous acknowledgment (continued)



continues on next page...

Figure 4.13: COREF disambiguates an ambiguous acknowledgment (continued)

```

 $i_{5,1} = \langle$  COREF : tacitNop[[User does say[no]]],
      COREF : pushRemind[COREF, User, past, refuseTaskAction, add-
      addToScene[e2_2]],
      COREF : command[User, addToScene[e2_2]],
      COREF : setPrag[inFocus(Y), inFocus(e2_2)]

```

An example from this user study in which COREF faces a richer kind of ambiguity about acknowledgments is given in Figure 4.14. The transcript of this dialogue is as follows:

```

COREF: the dark orange circle
COREF: did you add it?
COREF: add it
a13: ok
COREF: ok
COREF: [clicks the Continue (next object) button]

```

In this dialogue, a13 utters *ok* at time 4, and COREF perceives the *ok* as ambiguous between three interpretations. According to $i_{4,3}$, the user simply acknowledges COREF's reminder to *add it*. According to $i_{4,1}$ and $i_{4,2}$, on the other hand, the user contributes that they have added the object to their scene.⁵ COREF therefore spawns three threads of interpretation. At time 5, COREF responds by saying *ok* back to the user, viewing itself as acknowledging the user's *ok* (no matter which contribution the user's *ok* actually made). Next, at time 6, COREF decides to click the **Continue (next object)** button and move on to the next object. In taking this action, COREF does something which is not coherent, according to its task models, under the assumption that the user's original *ok* was a simple acknowledgment (**nop**). (In fact, the user here had already added an object to their scene.) This lack of coherence means that COREF must drop that thread of interpretation at time 6. This dynamic illustrates a new kind of risk, as well as a new kind

⁵The kind of ambiguity that COREF perceives between $i_{4,1}$ and $i_{4,2}$, which differ in that $i_{4,2}$ substitutes a **tacitAbandonTasks** action for a **tacitNop** action, occurred frequently in this user study. Intuitively, we can explain this as COREF being unable to tell whether the user has abandoned the **Remind** task that COREF has previously pushed, or as instead tacitly acknowledging it. However, it is often hard to see exactly what the difference between these two particular interpretations is, and as such, we view this ambiguity as exhibiting a limitation in COREF's current interpretation process.

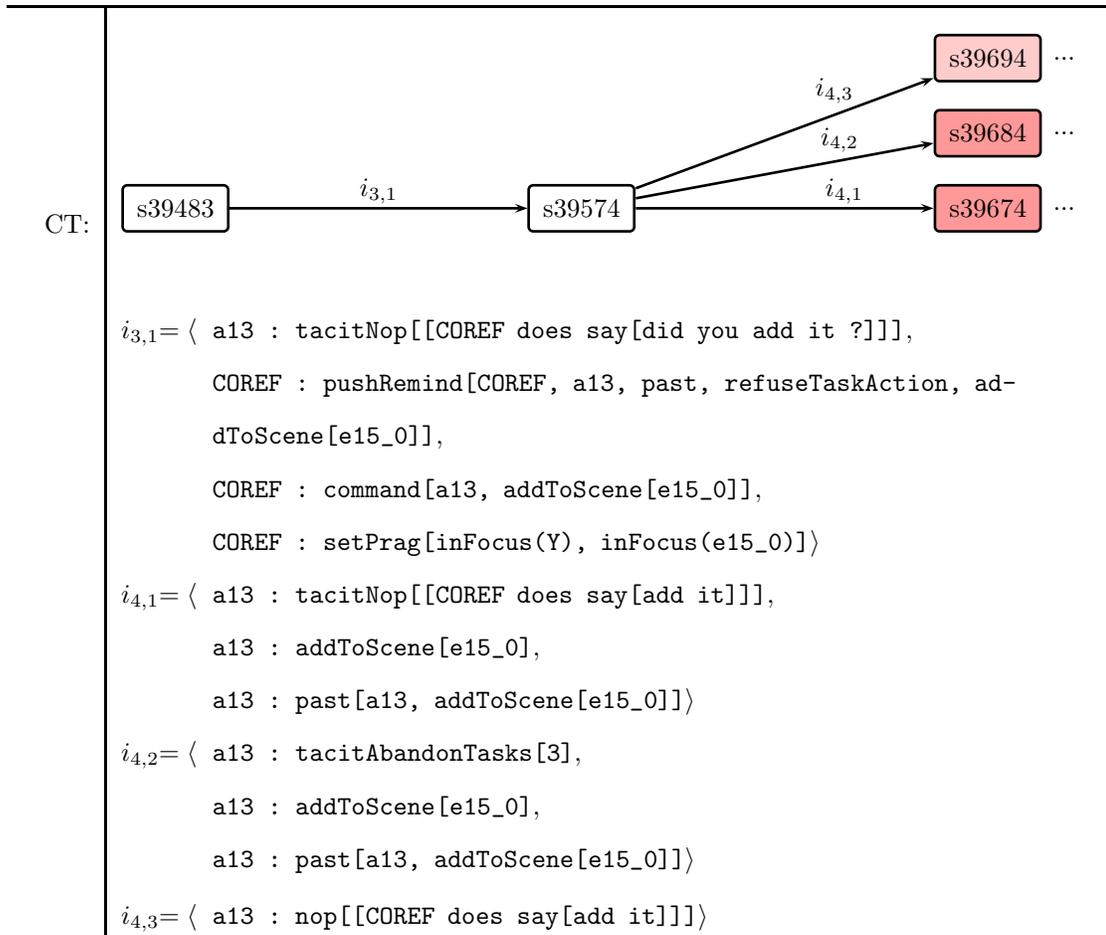
of flexibility, that contribution tracking affords to dialogue agents: they can assume a thread of coherent interpretation for the preceding dialogue that allows them to move forward with their task.

Figure 4.14: COREF proceeds despite an ambiguous acknowledgment

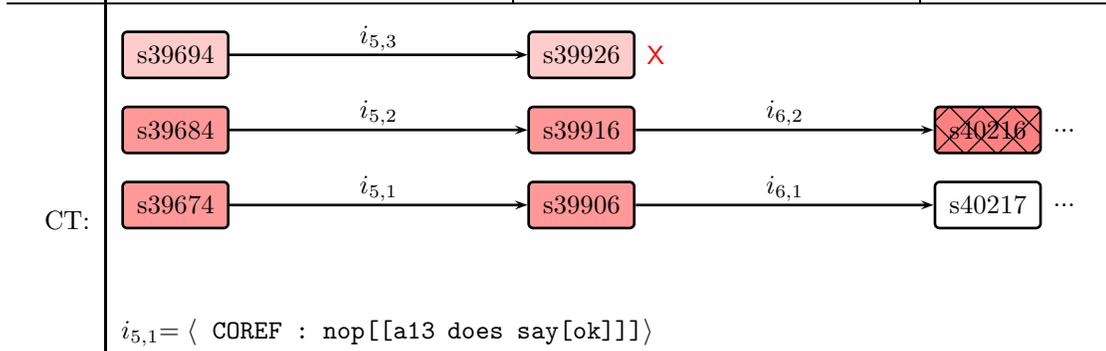
EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p> </div> <div style="text-align: center;"> <p>Your scene</p> </div> </div> <p>...</p>		
Time:	1	2	
SE:	COREF: <i>the dark orange circle</i>	COREF: <i>did you add it ?</i>	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s39051</div> $i_{1,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s39256</div> $i_{2,1}$ → <div style="border: 1px solid black; padding: 2px 10px;">s39483</div> ... </div> $i_{1,1} = \langle$ <ul style="list-style-type: none"> COREF : pushCollabRef[COREF, a13, t14], COREF : addcr[t14, equal(t14, e15_0)], COREF : setPrag[inFocus(Z), inFocus(t14)] \rangle $i_{2,1} = \langle$ <ul style="list-style-type: none"> a13 : tacitNop[[COREF does say[the dark orange circle]]], a13 : setVarValue[t14, e15_0], COREF : pushRemind[COREF, a13, past, refuseTaskAction, addToScene[e15_0]], COREF : askYNQ[past[a13, addToScene[e15_0]]], COREF : setPrag[inFocus(Y), inFocus(e15_0)] \rangle		
EI:	
Time:	3	4	
SE:	COREF: <i>add it</i>	a13: <i>ok</i>	

continues on next page...

Figure 4.14: COREF proceeds despite an ambiguous acknowledgment (continued)



EI:	
Time:	5	6	
SE:	COREF: ok	[COREF does clickContinue[]]	



continues on next page...

Figure 4.14: COREF proceeds despite an ambiguous acknowledgment (continued)

$i_{5,2}$	$= \langle \text{COREF} : \text{nop}[[\text{a13 does say}[\text{ok}]]] \rangle$
$i_{5,3}$	$= \langle \text{COREF} : \text{nop}[[\text{a13 does say}[\text{ok}]]] \rangle$
$i_{6,1}$	$= \langle \text{a13} : \text{tacitNop}[[\text{COREF does say}[\text{ok}]]],$ $\text{COREF} : \text{continueTask}[\text{t14}] \rangle$
$i_{6,2}$	$= \langle \text{a13} : \text{tacitNop}[[\text{COREF does say}[\text{ok}]]],$ $\text{COREF} : \text{continueTask}[\text{t14}] \rangle$

It is harder to capture and assimilate ambiguous acknowledgments into an incremental common ground model of dialogue (Clark and Schaefer, 1989; Traum, 1994) due to the special role that acknowledgments play in these models. These models attempt to track the material that is *grounded* at each step in a dialogue. When a new utterance occurs, its content may be marked *ungrounded* or *pending* to reflect the fact that its content must be acknowledged by the hearer before it can be assumed to have become common ground (Traum, 1994; Poesio and Traum, 1997). Perhaps the easiest way this approach can be squared with the existence of utterances which are ambiguous between an acknowledgment reading and another reading is by taking a classification approach to interpretation: in interpreting a user utterance, the utterance is judged either to be an acknowledgment or not to be one. It would not be straightforward, however, to ramify the system's probabilistic uncertainty about interpretation into this kind of dialogue model, because the uncertainty would seem to interfere with a binary classification of the acknowledged material as either *grounded* or *ungrounded*.

COREF therefore demonstrates a new qualitative capability for implemented dialogue systems that aim for collaborative language use: they can represent, reason about, and respond in collaborative ways to the ambiguity that accompanies acknowledgments in real-world, task-oriented dialogue.

4.3.2 Implementing ambiguity management as a flexible collaborative activity

Dialogue agents cannot always understand their human partners. Indeed, we ourselves do not always understand what others say to us. Nevertheless, *our* conversational abilities allow us to follow up provisional interpretations of what has been said and eventually arrive at a sufficient understanding.

There are currently two main approaches to designing dialogue agents that are similarly able to follow up on their uncertainty about user utterances and eventually figure out the user’s meaning. In this section, we review these existing approaches, and use example COREF subdialogues to present *collaborative ambiguity management* as a new strategy which dialogue agents can use to try to achieve this kind of competence in dialogues with their users.

In task optimization approaches to dialogue, the underlying dialogue model is often explicitly probabilistic; see Section 2.3.1 (page 16). This makes representing and reasoning about uncertainty in interpreting user utterances a natural strength of these models. For example, using speech recognition results to maintain a probability distribution over alternative user inputs can help a system to choose whether to clarify the user’s input or proceed with a possibly incorrect interpretation (e.g. Roy et al., 2000; Bohus and Rudnicky, 2006; Williams, 2008). It also allows statistical inference to combine evidence about user intentions from multiple utterances (Bohus and Rudnicky, 2006).

In order to provide a concrete example, we will take POMDP-based dialogue modeling as a representative of the task optimization approach. As discussed in Section 3.1.3 (page 68), POMDP-based dialogue models generally identify the dialogue state with the user’s private state or goal, and select system utterances and actions that optimize task performance under uncertainty about the state. In particular, in terms of managing perceived ambiguities, some of the types of system utterances that may be available are confirmation questions, clarification questions, or requests for the user to repeat their utterance (e.g. Roy et al., 2000; Williams, 2008). For example, the POMDP-based call center application of Williams (2008) supports dialogues like:

S: First and last name?
U: Junlan Feng
S: Sorry, first and last name?
U: Junlan Feng
S: Junlan Feng.
U: Yes
S: Dialing

In this interaction, the system *S* relies on its evolving uncertainty about the user *U*’s desired directory listing, along with other features, to decide whether to ask the user to repeat the name, to confirm a likely name, or simply to dial a specific name. As we argued in Section 3.1.3, this kind of POMDP model connects uncertainty to the system’s high-level choices in a way that facilitates task success, but because it focuses on modeling user state rather than dialogue context, it cannot

easily connect the system’s uncertainty to compositional, context-sensitive linguistic reasoning such as decision-making in natural language generation.

The second main approach draws on an incremental common ground (ICG) representation to track the evolving dialogue context; see Section 2.3.2.3 (page 23). Systems based on ICG rely on relatively detailed models of the evolving dialogue context to interpret and generate context-sensitive linguistic constructions at each point in the dialogue. When compared with POMDP approaches, these models are less conducive to straightforward probabilistic reasoning, but go much further in accounting for the specific utterances speakers can use recognizably in particular contexts. Work in the ICG tradition also includes more detailed models of the various kinds of grounding and clarification utterances that interlocutors can use to follow up on their uncertainty about previous utterances (e.g. Traum, 1994; Ginzberg and Cooper, 2004; Purver, 2004).

The commitment to achieving common ground at each step, in the ICG approach, has led many systems and dialogue models to delay context update after each utterance until any perceived ambiguities have been resolved (Ginzberg and Cooper, 2004; Purver, 2004), and in some cases until an explicit grounding action has occurred after each utterance (Traum, 1994). We now draw on COREF’s contribution tracking to demonstrate a new approach which maintains the collaborative view of language that has motivated much work in the ICG tradition, but which decouples context update from the resolution of perceived ambiguities. We call this new approach *collaborative ambiguity management*.

Collaborative ambiguity management is a dialogue strategy that draws on contribution tracking to allow dialogue agents to flexibly and robustly address the general problem of responding to perceived ambiguities in dialogue. One of the motivations for this approach is that the problem of recognizing and responding to perceived ambiguities in a collaboration seems to be more general than many of the specific problems associated with modeling clarification and grounding subdialogues. For example, in the human-human subdialogue depicted in Figure 4.12 (page 168), the question *so you’ve added it?* serves to resolve ambiguity just like a clarification might, but the question is not expressed using any special syntactic form that is characteristic of follow-up utterances (Fernández, 2006), and nor does the pragmatic interpretation of this question seem to require any clarification-specific context update mechanism (Purver, 2004). Rather, this just seems to be a run-of-the-mill question, but one which has been asked in response to a perceived ambiguity, and one whose answer is expected to help resolve that ambiguity.

We call such a question, which is asked by an agent who is uncertain about the dialogue context due to a previously perceived ambiguity, and with the motivation to reduce the agent’s uncertainty,

an *ambiguity management question* (AMQ). We take the position that ambiguity management plays out as part of a broader collaboration; as such, AMQs, just like other types of questions, and indeed utterances more generally, should be *contributive*. Further, AMQs fit into a broader framework of ambiguity management in which contribution tracking gives agents the flexibility, in some situations, to proceed despite a perceived ambiguity, and to formulate contributive AMQs that may help resolve ambiguity when so desired.

Collaborative ambiguity management and ICG models tell different stories about the decision-making problem faced by an interlocutor who has found multiple competing interpretations for an utterance. In an ICG model, the primary question is *what can I assume is common ground at this point?* In collaborative ambiguity management, the question is rather *how many competing threads of interpretation am I willing to tolerate going forward?* The difference in these two questions potentially provides extra flexibility to a dialogue agent that implements collaborative ambiguity management rather than an ICG model.

COREF implements collaborative ambiguity management by formulating *all* its questions to be contributive, including its clarification questions and other AMQs, and using a specific question-asking policy. COREF's hand-built question-asking policy decides which questions would be acceptable for the agent to ask given its current uncertainty. For example, COREF's policy deems it acceptable to ask for clarification any time COREF is uncertain which constraint a speaker intended to add with an utterance, as occurs at time 2 in Figure 4.8 (page 154). Similarly, COREF's question-asking policy deems it acceptable for the agent to ask whether a non-public action m has occurred, if some possible contexts but not others indicate that m has taken place. Such a question occurs when COREF asks *did you add it?* in the interaction of Figure 3.3 (page 81) in Chapter 3.

To help illustrate COREF's contributive question-asking in this user study, we show all the questions COREF asked in this user study in Figure 4.15.

Figure 4.15: All questions asked by COREF in the user study

count	question	count	question
159	<i>did you add it?</i>	21	<i>is the target round?</i>
12	<i>did you add the circle?</i>	18	<i>is the target solid?</i>
7	<i>did you add the square?</i>	16	<i>is the target square?</i>
6	<i>do you mean dark brown?</i>	2	<i>is the target the beige circle?</i>
1	<i>do you mean dark orange?</i>	2	<i>is the target the beige diamond?</i>

continues on next page...

Figure 4.15: All questions asked by COREF in the user study
(continued)

count	question	count	question
2	<i>do you mean it?</i>	2	<i>is the target the beige square?</i>
1	<i>do you mean the beige square?</i>	9	<i>is the target the blue circle?</i>
1	<i>do you mean the light blue circle?</i>	13	<i>is the target the blue diamond?</i>
1	<i>do you mean the light blue object?</i>	10	<i>is the target the blue square?</i>
1	<i>do you mean the pink circle?</i>	16	<i>is the target the brown circle?</i>
1	<i>do you mean the solid beige diamond?</i>	5	<i>is the target the brown diamond?</i>
1	<i>do you mean the solid circle?</i>	3	<i>is the target the brown square?</i>
1	<i>do you mean the solid pink circle?</i>	9	<i>is the target the circle?</i>
1	<i>do you mean the solid square?</i>	1	<i>is the target the dark blue circle?</i>
3	<i>i mean?</i>	2	<i>is the target the dark blue diamond?</i>
8	<i>is it beige?</i>	5	<i>is the target the dark blue square?</i>
1	<i>is it brown?</i>	1	<i>is the target the dark brown circle?</i>
5	<i>is it empty?</i>	1	<i>is the target the dark orange circle?</i>
2	<i>is it pink?</i>	2	<i>is the target the dark orange diamond?</i>
6	<i>is it round?</i>	9	<i>is the target the diamond?</i>
9	<i>is it solid?</i>	3	<i>is the target the empty brown circle?</i>
7	<i>is it square?</i>	2	<i>is the target the empty brown object?</i>
1	<i>is the circle an object?</i>	3	<i>is the target the empty circle?</i>
2	<i>is the circle dark blue?</i>	5	<i>is the target the empty diamond?</i>
1	<i>is the circle dark orange?</i>	1	<i>is the target the empty pink square?</i>
1	<i>is the circle light blue?</i>	4	<i>is the target the empty square?</i>
1	<i>is the circle the brown square?</i>	1	<i>is the target the light blue circle?</i>
1	<i>is the circle the dark blue square?</i>	1	<i>is the target the light blue diamond?</i>
1	<i>is the circle the light blue square?</i>	2	<i>is the target the light blue square?</i>
1	<i>is the circle the solid diamond?</i>	1	<i>is the target the orange circle?</i>
1	<i>is the diamond pink?</i>	1	<i>is the target the orange square?</i>
1	<i>is the diamond the object?</i>	1	<i>is the target the pink circle?</i>
1	<i>is the object the dark orange circle?</i>	5	<i>is the target the pink diamond?</i>

continues on next page...

Figure 4.15: All questions asked by COREF in the user study
(continued)

count	question	count	question
1	<i>is the square dark blue?</i>	1	<i>is the target the pink square?</i>
26	<i>is the target a diamond?</i>	2	<i>is the target the solid beige diamond?</i>
5	<i>is the target beige?</i>	1	<i>is the target the solid beige square?</i>
24	<i>is the target blue?</i>	1	<i>is the target the solid blue square?</i>
3	<i>is the target brown?</i>	1	<i>is the target the solid brown diamond?</i>
7	<i>is the target dark blue?</i>	4	<i>is the target the solid circle?</i>
8	<i>is the target dark brown?</i>	1	<i>is the target the solid dark blue circle?</i>
17	<i>is the target dark orange?</i>	1	<i>is the target the solid object?</i>
1	<i>is the target empty?</i>	1	<i>is the target the solid orange square?</i>
3	<i>is the target light blue?</i>	2	<i>is the target the solid square?</i>
2	<i>is the target pink?</i>	22	<i>is the target the square?</i>

COREF viewed all these utterances as contributive given the uncertainty (or certainty) it faced at the time it asked the question. The subtle variation in lexical choices between these questions reflects COREF's varying assessments of the expected recognizability of the contribution it wants to make. For example, a choice of *did you add it?* rather than *did you add the circle?* (or *did you add the square?*) generally depends on COREF's assessment about whether the object COREF wants to refer to is in focus in (all) the relevant context(s). A similar determination guides COREF's selection of *is it round?* rather than *is the target round?* in particular dialogue scenarios.

The same kind of contributivity assessment guides COREF's generation process in selecting clarification questions. For example, at time 3 in the object subdialogue of Figure 4.8 (page 154), COREF asks *do you mean dark brown?*. Figure 4.16 depicts SPUD's search process for this utterance. Note in particular that SPUD considers simply generating the question *do you mean brown?* before settling on *do you mean dark brown?* It rejects the former utterance because *do you mean brown?* is not expected to be recognized as unambiguously contributing COREF's desired dialogue act (i), because it would also support interpretations (ii):

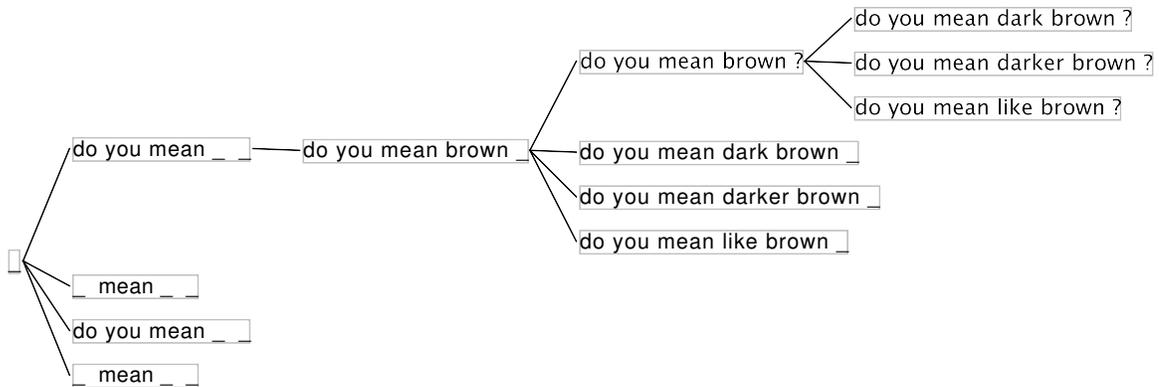


Figure 4.16: SPUD generates *do you mean dark brown?* for COREF. In this figure, each search node considered by SPUD is depicted using only the surface text at that search node. Nodes are ranked vertically, with higher ranked nodes above lower ranked nodes. An underscore indicates the presence of a syntactic gap in the provisional syntactic structure at the node.

- (i) `askYNQ[addcr[Brown, equals(Brown, saddleBrownFigureObject)]]`
- (ii) `askYNQ[addcr[Brown, equals(Brown, sandyBrownFigureObject)]]`

In this way, as part of its search for a contributive clarification question to ask, COREF avoids using a simple word like *brown* when COREF itself would (in fact, did!) perceive that word as ambiguous in the current context.

COREF’s determination to speak contributively is a productive capacity that transfers to all the questions depicted in Figure 4.15, whether they are clarification questions or not, and whether they were asked at a time when COREF was uncertain about the context or not. One attractive aspect of COREF’s collaborative ambiguity management, then, is that the agent relies on the same reasoning process of contribution tracking to formulate all the different kinds of questions, and utterances more generally, that it can produce.

Another attractive aspect of COREF’s collaborative ambiguity management is that it provides a relatively flexible and robust approach to responding to perceived ambiguities. By *flexible*, we mean that disambiguating information can come in a variety of forms. By *robust*, we mean the information carried by an utterance is not forgotten, “downdated”, or left in a pending status if an attempt to clarify fails. We now give a few examples that illustrate these features.

One outcome that one of COREF’s AMQs can have is for the ambiguity to be completely resolved by the user’s answer. We have seen several examples of this kind already. One example is the clarification question *do you mean dark brown?* which COREF asks in the interaction of Figure

4.8 (page 154). We give another example from this user study, in which a clarification question successfully resolves a perceived ambiguity, in Figure 4.17. Together with successful clarification questions, we can also include Figure 4.13 (page 169) as another type of interaction in which COREF is able to eliminate a perceived ambiguity using an AMQ; this time COREF is asking *did you add it?* rather than an explicit clarification question. These kinds of examples could be handled in an ICG-based dialogue system as well; in particular, the successful resolution of the agent's perceived ambiguity can be narrated as justifying an assumption that the disambiguated contribution has become common ground, which allows the dialogue context to be safely updated to reflect the contribution.

Figure 4.17: COREF's clarification question eliminates a perceived ambiguity

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p> </div> <div style="text-align: center;"> <p>Your scene</p> </div> </div> <p>...</p>		
Time:	1	2	
SE:	COREF: <i>is the target pink ?</i>	a18: <i>no</i>	
CT:	<div style="display: flex; align-items: center; gap: 20px;"> <div style="border: 1px solid black; padding: 2px 10px;">s55812</div> <div style="text-align: center;">$i_{1,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s55979</div> <div style="text-align: center;">$i_{2,1}$</div> <div style="border: 1px solid black; padding: 2px 10px;">s57195</div> <div>...</div> </div> <p> $i_{1,1} = \langle$ COREF : tacitNop[[a18 does clickContinue[]]], a18 : pushCollabRef[a18, COREF, t18], COREF : pushYNQ[COREF, a18, addcr[t18, P], negcr[t18, P]], COREF : askYNQ[addcr[t18, palevioletredFigureObject(t18)]], COREF : setPrag[inFocus(Y), inFocus(t18)] \rangle </p> <p> $i_{2,1} = \langle$ a18 : tacitNop[[COREF does say[is the target pink ?]]], a18 : negcr[t18, palevioletredFigureObject(t18)] \rangle </p>		
EI:	

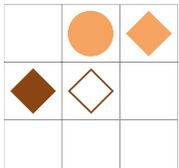
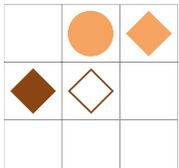
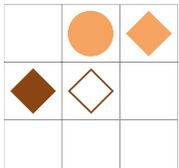
continues on next page...

Figure 4.17: COREF's clarification question eliminates a perceived ambiguity (continued)

Time:	3	4	
SE:	a18: <i>it is brown</i>	COREF: <i>do you mean dark brown ?</i>	
CT:	<pre> graph LR n1[s57195] -- i3,2 --> n2[s58148] n2 -- i4,2 --> n3[s58416] n1 -- i3,1 --> n4[s58133] n4 -- i4,1 --> n5[s58406] n3 --- dots1[...] n5 --- dots2[...] </pre> <p> $i_{3,1} = \langle \text{COREF : tacitNop}[[\text{a18 does say}[\text{no}]]],$ $\text{a18 : addcr}[\text{t18, saddlebrownFigureObject}(\text{t18})],$ $\text{a18 : setPrag}[\text{inFocus}(\text{Y}), \text{inFocus}(\text{t18})]\rangle$ $i_{3,2} = \langle \text{COREF : tacitNop}[[\text{a18 does say}[\text{no}]]],$ $\text{a18 : addcr}[\text{t18, sandybrownFigureObject}(\text{t18})],$ $\text{a18 : setPrag}[\text{inFocus}(\text{Y}), \text{inFocus}(\text{t18})]\rangle$ $i_{4,1} = \langle \text{COREF : pushClarify}[[\text{a18 does say}[\text{it is brown}]], \text{a18, COREF,}$ $\text{Brown}],$ $\text{COREF : pushYNQ}[\text{COREF, a18, addcr}[\text{Brown, P}], \text{negcr}[\text{Brown, P}]],$ $\text{COREF : askYNQ}[\text{addcr}[\text{Brown, equal}(\text{Brown, saddlebrownFigureOb-}$ $\text{ject})]]\rangle$ $i_{4,2} = \langle \text{COREF : pushClarify}[[\text{a18 does say}[\text{it is brown}]], \text{a18, COREF,}$ $\text{Brown}],$ $\text{COREF : pushYNQ}[\text{COREF, a18, addcr}[\text{Brown, P}], \text{negcr}[\text{Brown, P}]],$ $\text{COREF : askYNQ}[\text{addcr}[\text{Brown, equal}(\text{Brown, saddlebrownFigureOb-}$ $\text{ject})]]\rangle$ </p>		
EI:	
Time:	5	6	
SE:	a18: <i>yes</i>	COREF: <i>done</i>	

continues on next page...

Figure 4.17: COREF's clarification question eliminates a perceived ambiguity (continued)

CT:	<div style="display: flex; align-items: center; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 5px;">s58416</div> ✗ </div> <div style="display: flex; align-items: center; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 10px;">s58406</div> $i_{5,1}$ s60204 $i_{6,1}$ s60320 ... </div> <p>$i_{5,1} = \langle$ a18 : tacitNop[[COREF does say[do you mean dark brown ?]]], a18 : addcr[Brown, equal(Brown, saddlebrownFigureObject)] \rangle</p> <p>$i_{6,1} = \langle$ COREF : tacitNop[[a18 does say[yes]]], COREF : setVarValue[Brown, saddlebrownFigureObject], COREF : setVarValue[t18, rh656899_4], COREF : addToScene[rh656899_4], COREF : past[COREF, addToScene[rh656899_4]] \rangle</p>				
EI:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: left;">Candidate Objects</th> <th style="width: 50%; text-align: left;">Your scene</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">  </td> <td style="text-align: center;">  </td> </tr> </tbody> </table>	Candidate Objects	Your scene		
Candidate Objects	Your scene				
					
Time:	7	8			
SE:	[perceived 8 figure objects]	COREF: <i>click continue</i>			
CT:	<div style="display: flex; align-items: center; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 10px;">s60320</div> $i_{7,1}$ s60325 $i_{8,1}$ s60390 ... </div> <p>$i_{7,1} = \langle$ COREF : perceive[PerceivedFigureObjectsEvent<[rh656899_4, e656905_4, ...]] \rangle</p> <p>$i_{8,1} = \langle$ a18 : tacitNop[[COREF does say[done]]], COREF : pushRemind[COREF, a18, past, refuseTaskAction, continue- Task[t18]], COREF : command[a18, continueTask[t18]] \rangle</p>				

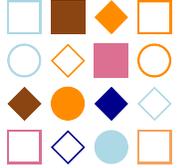
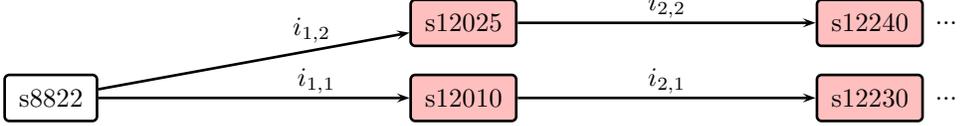
Connecting context update to the resolution of perceived ambiguities may guarantee common ground, but leaving ambiguities open can make a collaborative agent more flexible. An agent that demands a clear context but lacks the resources to clarify something may have no recourse but to take a “downdate” action—to signal to the user that their intended contribution was not understood, and discard any alternative possible contents. If the agent can proceed, however, the agent may get evidence from what happens next to resolve its uncertainty and complete the task.

In COREF’s case, we have already seen that COREF’s AMQs do not always eliminate a perceived ambiguity. In Chapter 3, we discussed the example of Figure 3.3, in which COREF’s question *did you add it?* does not reduce the number of contexts which COREF sees as viable. COREF nevertheless is able to proceed and achieve a *correct* object outcome in that example. The example of Figure 4.9 (page 159) provides another instance in which an AMQ does not eliminate a perceived ambiguity; here COREF’s poorly worded AMQ *do you mean it?* prevents the user’s answer from reducing the number of viable contexts COREF perceives.

An important qualitative feature of collaborative ambiguity management is that it allows an agent to keep talking and try to complete the task even when AMQs do not completely eliminate the ambiguity. For example, Figure 4.18 shows COREF’s implemented contribution tracking during a hypothetical clarification attempt. In this interaction, the user begins by saying *the target is orange*. COREF sees the word *orange* as ambiguous, and responds with an AMQ: *do you mean dark orange?* The user here responds not by answering COREF’s question, but rather by saying *it’s an empty circle*. Given the array of visible objects, this response is actually a reasonably efficient way to complete the task: there is only one object that could be described as both *orange* and *an empty circle*. In this example, COREF is able to interpret the user’s response *it’s an empty circle* as abandoning COREF’s clarification question, and adds two additional constraints – that the target is empty and a circle – as extensions to each of its two threads of interpretation. At time 4, by considering these threads, COREF is able to determine that only one thread of interpretation is internally consistent, and so COREF adds the dark orange empty circle to its scene and instructs the user to continue on to the next object. COREF’s ambiguity is resolved. An important aspect of this example is that COREF needs to aggregate information both from the ambiguous utterance and from information that follows its failed clarification attempt in order to complete the task successfully. This kind of flexibility seems less easy to implement on an ICG approach, because in principle, if a clarification attempt fails, then common ground regarding the to-be-clarified material is not achieved, and so this material should be forgotten, downdated, or remain in a pending status

when the dialogue proceeds.⁶

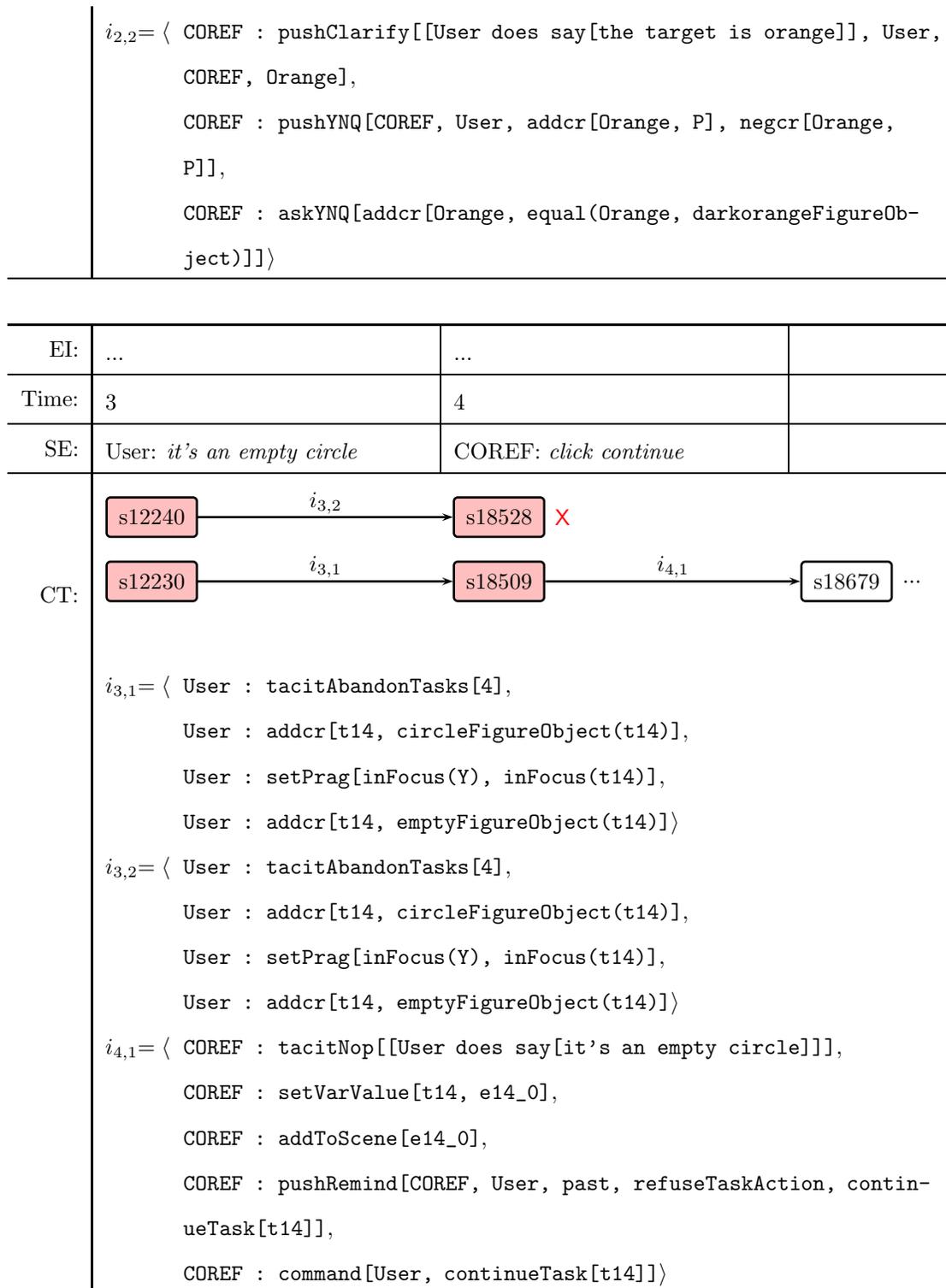
Figure 4.18: COREF's uncertainty is resolved flexibly

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div>	...	
Time:	1	2	
SE:	User: <i>the target is orange</i>	COREF: <i>do you mean dark orange ?</i>	
CT:	 <pre> graph LR s8822[s8822] -- i1,1 --> s12010[s12010] s8822 -- i1,2 --> s12025[s12025] s12010 -- i2,1 --> s12230[s12230] s12025 -- i2,2 --> s12240[s12240] </pre> <p> $i_{1,1} = \langle$ User : pushCollabRef [User, COREF, t14], User : addcr [t14, darkorangeFigureObject (t14)], User : setPrag [inFocus (Y), inFocus (t14)]\rangle </p> <p> $i_{1,2} = \langle$ User : pushCollabRef [User, COREF, t14], User : addcr [t14, sandybrownFigureObject (t14)], User : setPrag [inFocus (Y), inFocus (t14)]\rangle </p> <p> $i_{2,1} = \langle$ COREF : pushClarify [[User does say [the target is orange]], User, COREF, Orange], COREF : pushYNQ [COREF, User, addcr [Orange, P], negcr [Orange, P]], COREF : askYNQ [addcr [Orange, equal (Orange, darkorangeFigureOb- ject)]]\rangle </p>		

continues on next page...

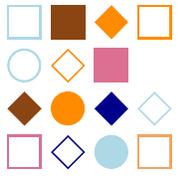
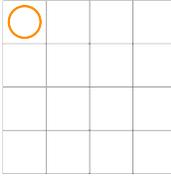
⁶It could be argued that this is not a failed clarification attempt at all, but rather a successful one: the user answers COREF's question *do you mean dark orange?* by asserting a proposition that implies the answer. The flexibility we are demonstrating here, however, does not depend on how this particular example is understood: COREF's contribution tracking allows it to aggregate information that comes from an ambiguous utterance with information that arrives several turns later and finally resolves the ambiguity. This is only possible if the agent remembers the ambiguity that was present in the original ambiguous utterance.

Figure 4.18: COREF's uncertainty is resolved flexibly (continued)



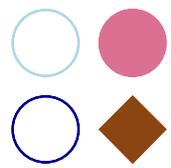
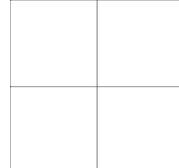
continues on next page...

Figure 4.18: COREF's uncertainty is resolved flexibly (continued)

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div>		
Time:	5		
SE:	[perceived 16 figure objects]		
CT:	<div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 2px 10px;">s18679</div> <div style="margin: 0 10px;"> $\xrightarrow{i_{5,1}}$ </div> <div style="border: 1px solid black; padding: 2px 10px;">s18684</div> <div style="margin-left: 5px;">...</div> </div> <p style="margin-top: 10px;">$i_{5,1} = \langle \text{COREF : perceive}[\text{PerceivedFigureObjectsEvent}\langle [r17_0, r25_0, rh20_0, r\dots] \rangle \rangle$</p>		

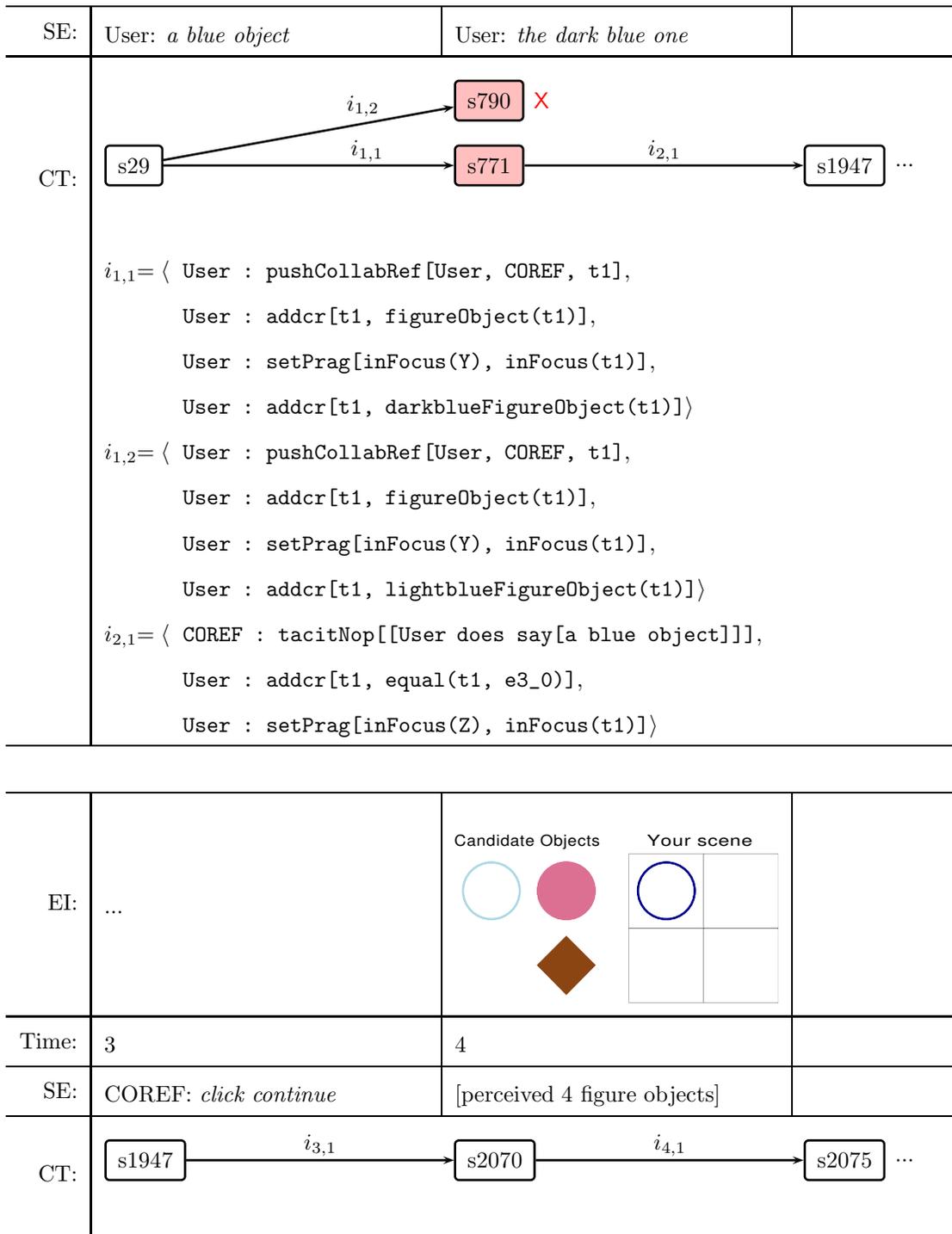
Another reason it can be advantageous to defer the resolution of a perceived ambiguity is that the user may disambiguate their own intention without specific intervention by the system. For example, Figure 4.19 shows COREF's implemented contribution tracking during a hypothetical interaction that illustrates this possibility. In this interaction, the user starts out by saying *a blue object*, which COREF perceives as ambiguous due to the presence of objects of two shades of blue in the display. Before COREF responds, the user continues by saying *the dark blue one*. This effectively eliminates COREF's perceived ambiguity without any specific effort apart from COREF's usual contribution tracking.

Figure 4.19: COREF's uncertainty is resolved flexibly

EI:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Candidate Objects</p>  </div> <div style="text-align: center;"> <p>Your scene</p>  </div> </div>	...	
Time:	1	2	

continues on next page...

Figure 4.19: COREF's uncertainty is resolved flexibly (continued)



continues on next page...

Figure 4.19: COREF's uncertainty is resolved flexibly (continued)

```

 $i_{3,1} = \langle$  COREF : tacitNop[[User does say[the dark blue one]],
    COREF : setVarValue[t1, e3_0],
    COREF : addToScene[e3_0],
    COREF : pushRemind[COREF, User, past, refuseTaskAction, continueTask[t1]],
    COREF : command[User, continueTask[t1]] $\rangle$ 
 $i_{4,1} = \langle$  COREF : perceive[PerceivedFigureObjectsEvent<[e3_0, e2_0, rh1_0, e0_0]>] $\rangle$ 

```

Taken together, these examples demonstrate a new qualitative capability for dialogue systems: we can use contribution tracking to implement ambiguity management as a flexible collaborative activity. This approach may help us to design agents that are more flexible dialogue partners that can keep talking and resolve their uncertainty over time, as needed.

We now turn to the quantitative capabilities we are able to articulate using this user study. In Section 4.4.1, we continue the theme of this section by quantifying COREF's ability to reduce its perceived ambiguity using ambiguity management questions.

4.4 New quantitative capabilities

In this section, we discuss the new quantitative capabilities that we have achieved in our implementation of contribution tracking in COREF.

4.4.1 Quantifying performance in collaborative ambiguity management

While certainty about the context is not strictly necessary for a *correct* outcome (Section 4.2), COREF nevertheless does often try to reduce its uncertainty according to its question-asking policy. In Section 4.3.2, we characterized COREF's approach to perceived ambiguities as one of *collaborative ambiguity management*, and discussed its use of ambiguity management questions (AMQs) to try to reduce its uncertainty in specific circumstances. In this section, we quantify the observed effectiveness of COREF's AMQs in this user study.

For the purposes of this analysis, we define an AMQ as a question asked by COREF at a moment when COREF was entertaining more than one thread of interpretation for the dialogue (or,

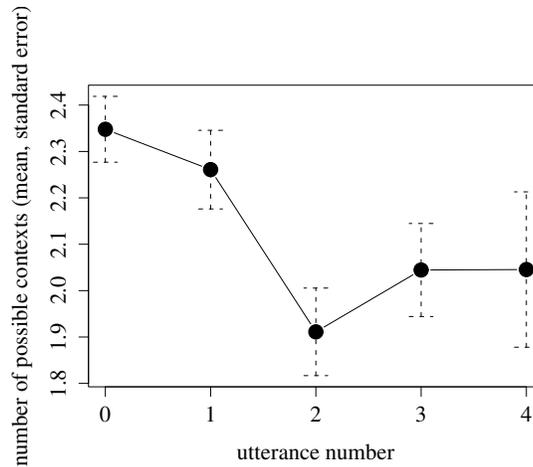


Figure 4.20: Effect of ambiguity management questions on COREF’s uncertainty. At utterance 0, COREF faces an ambiguous context. At utterance 1, COREF has asked a question. Typically, at utterance 2, the user has answered COREF’s question.

equivalently, at a moment when COREF viewed more than one dialogue state as viable). Under this definition, several kinds of questions qualify as AMQs. These include responses to ambiguous acknowledgments, such as *did you add it?* in Figure 3.3 (page 81). This definition also counts explicit clarification questions, such as *do you mean dark brown?* in Figure 4.8 (page 154), as AMQs. Finally, more general yes/no questions such as *is the target the beige circle?* are also counted as AMQs if COREF asks them while uncertain about the dialogue state. Our definition of AMQs thus reflects a general orientation towards question-asking under uncertainty, rather than towards specific categories of questions, such as clarifications.

In total, COREF asked 46 AMQs in the user study. This amounts to 2.3 AMQs per subject, on average. These 46 AMQs occurred during 41 (7.1%) of the 580 object subdialogues in the study. Thus, most object subdialogues did not include an AMQ. On a few occasions, COREF asked more than one AMQ during a single object subdialogue.

Figure 4.20 illustrates the effectiveness of COREF’s question-asking policy at reducing uncertainty. As the figure shows, when COREF asks questions in an ambiguous context, the mean reduction in the agent’s uncertainty is about 0.4 contexts. The variation in uncertainty reduction reflects the fact that COREF’s contribution tracking admits a number of different outcomes after a question is asked: the user’s answer may eliminate one or more contexts; the user’s answer may not eliminate any contexts; the user’s answer may itself be ambiguous and *increase* the agent’s uncertainty; the user may choose not to answer the question; or the agent may fail to understand the user’s answer.

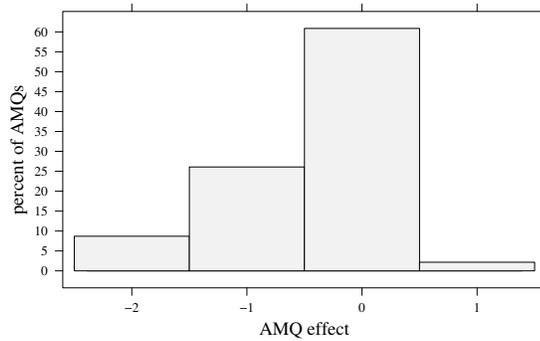


Figure 4.21: Distribution of AMQ effects for COREF's AMQs.

COREF's contribution tracking allows it to execute a uniform update to its uncertainty for all of these different types of outcomes.

In Figure 4.20, the effect of COREF's AMQs can be approximately identified with the change in uncertainty that occurs between time 0, before COREF asks the question, and time 2, which is typically the time at which the user has just answered the question. We shall call this change in the number of viable contexts at time 0 and at time 2 the *AMQ effect*. Since COREF entertains exactly 1, 2, or 3 contexts as viable at each point in time, the AMQ effect $e = |v_2| - |v_0|$ necessarily lies in the range $-2 \leq e \leq 2$. A negative AMQ effect represents a reduction in the agent's uncertainty.

Figure 4.21 shows the distribution in AMQ effects for COREF's AMQs in this user study. The figure shows that the most common outcome, occurring for 62.2% of COREF's AMQs, is for no change in uncertainty to occur from time 0 to time 2. This largely reflects the frequent occurrence of *did you add it?* AMQs, to which the user generally responds *yes*, as illustrated in Figure 3.3 (page 81) in Chapter 3. (When COREF asks *did you add it?*, the user has generally already added the object; however, even if they have not already added the object, the user usually tacitly adds the object and responds *yes*, rather than responding *no* to COREF's question.) In such AMQs, the agent's uncertainty is not reduced, but on the other hand, the agent does become certain that the object has been added, which allows it to decide to continue on to the next object with lower risk of a *no object* outcome.

For 35.6% of COREF's AMQs, the AMQ effect is -1 or -2, representing an elimination of 1 or 2 contexts, respectively. This category includes successful clarification questions such as the one in Figure 4.8 (page 154). This analysis therefore allows us to quantify COREF's ability to successfully reduce its uncertainty by 1 or more contexts, when it decides to attempt to do so by asking an AMQ,

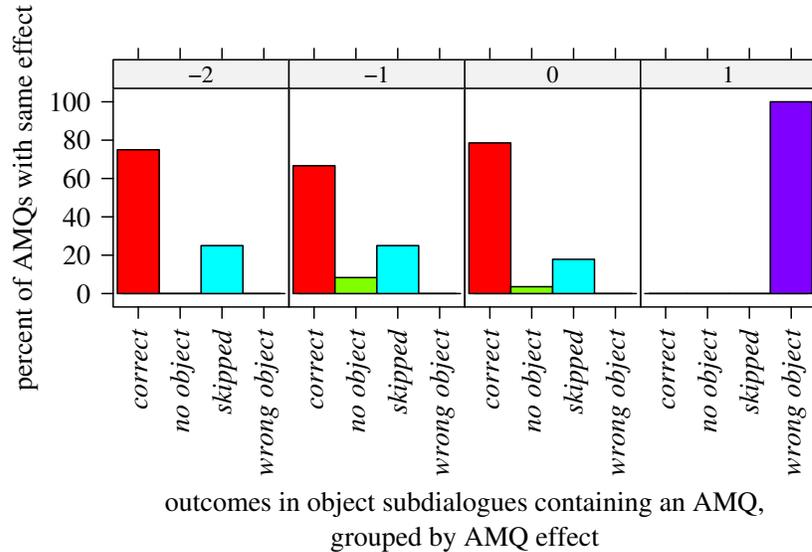


Figure 4.22: Relation between object outcome and AMQ effect. This figure shows the outcome that followed each AMQ asked by COREF. These outcomes are grouped according to the effect of the relevant AMQ. The leftmost bin, for AMQ effect -2, represents 4 AMQs. The bin for effect -1 represents 12 AMQs. The bin for effect 0 represents 28 AMQs. The bin for effect 1 includes 1 AMQ.

at 35.6%.

The remaining 2.2% of COREF’s AMQs comprise a single AMQ whose effect was to increase COREF’s uncertainty by 1 context.

Because this user study was not designed to provide a large number of paired examples in which COREF both did and did not ask an AMQ, it is difficult to develop a causal explanation of COREF’s object outcomes in terms of its AMQ policy. Figure 4.22 presents a descriptive picture, however, of how the effect of an AMQ related to object outcomes in this user study. Qualitatively, the figure seems to show relatively similar object outcomes for objects which led COREF to ask AMQs whose effects were -2, -1, or 0. It should not be inferred from this data, however, that object outcome is unaffected by COREF’s AMQs. One reason is the relatively small number of AMQs represented in the figure. Another reason is that successful clarification questions, such as COREF’s AMQ (with effect -1) *do you mean dark brown?* in Figure 4.8 (page 154), clearly play a causal role in achieving a *correct* object outcome. More generally, in this particular study, we lack a control condition to capture what outcome would have occurred if an AMQ had not been asked, or if the same AMQ had been asked but a different user answer had had a different effect on the agent’s uncertainty. One aspect that would need to be controlled for, for example, is the initial uncertainty before an AMQ

is asked. (Note that the bins in Figure 4.22 are not normalized with respect to initial uncertainty.) Finally, Figure 4.5 (page 148) suggests that the reduction in COREF’s uncertainty associated with AMQs of effect -1 or -2 should in fact lead to better object outcomes, especially in situations in which COREF faces high initial uncertainty.

Taken together, Figures 4.5 (page 148), 4.20 (page 190), 4.21 (page 191), and 4.22 (page 192) suggest that COREF’s approach to ambiguity management is relatively successful in cases of mild or short-lived ambiguities.

The type of analysis we have presented here, which quantifies the effect of an agent’s utterances on its own evolving uncertainty about the dialogue state, is closely related to the quantitative modeling that occurs in POMDP-based dialogue models. For example, Roy et al. (2000) investigate the reduction in the entropy in the system’s belief state that occurs when the system asks a confirmation or clarification question following an ambiguous user utterance. What’s new here is that our approach fully integrates COREF’s ambiguity-related decision-making with its collaborative reasoning about language use in context.

4.4.2 Leveraging contribution tracking to tune an agent’s interpretation model

An agent that implements contribution tracking has some ability to tolerate ambiguity as it moves forward with its dialogues, and some ability to resolve its ambiguity over time. The resolution of ambiguity may involve some combination of asking questions of the user, aggregating information provided by the user across multiple turns of dialogue, or strategically dropping threads of interpretation when the agent judges it can no longer sustain them. In the case of COREF, we have seen that COREF uses contribution tracking to entertain up to three alternative threads of interpretation, and that it uses a specific question-asking policy to try to reduce its uncertainty in specific situations. In this section, we show that this general perspective on ambiguity management through contribution tracking provides an agent with a source of information that can be used to tune its interpretation models to better fit its own dialogue experience.

As discussed in Section 3.3.6.1 (page 105), COREF’s Sensory Event Interpreter draws on a number of factors to construct interpretations for its sensory events in particular dialogue states. These factors include features of specific dialogue states, including: the stack of tasks that are underway in the dialogue state, the specific tacit action sequences that are judged coherent in the horizon graph, and the propositions that have previously been added to the dialogue state,

among other things. These factors also include various features involved in analyzing the specific utterance or action that is being interpreted, including: the words that have been used, alternative interpretations for referring expressions, alternative word senses (e.g. for color terms like *brown*), and alternative syntactic and semantic analyses of the utterance, among other things.

In this setting, we view COREF’s Sensory Event Interpreter as making a number of assumptions in order to construct each interpretation. Because multiple interpretations may be viable, the agent must somehow reconcile or weigh the relative likelihood of each interpretation. COREF’s architecture formalizes this assessment in a step, described in Section 3.3.6.1.3 (page 116), in which the agent assigns a probability $P(I = i_{t,j}|o, S_t = s_k)$ to each interpretation $i_{t,j}$ that is hypothesized as an explanation for sensory event o under the assumption that the dialogue state S_t at time t is s_k . In general, this step assesses a set of interpretations $\{i_{t,1}, \dots, i_{t,n}\}$ for each event o and assumed state s_k .

In the user-agent interactions in this user study, COREF assigned these probabilities using the hand-built model discussed in Section 3.3.6.1.3:

$$P(I = i_{t,j}|o, S_t = s_k) \propto \frac{1}{\text{NUM-TACIT-ACTIONS}(i_{t,j}) + 1} \quad (4.1)$$

This model essentially ignores most of the factors mentioned above, and simply weighs the relative likelihoods of alternative interpretations wholly in terms of the relative numbers of tacit actions in the interpretations. This constitutes a gross simplification, however, of the abductive inference process (Hobbs et al., 1993; Thomason et al., 2006) that we believe should determine this probabilistic assessment. For example, the hand-built model ignores alternative interpretations for the sense of color terms like *brown*, which in COREF’s domain model can be associated with several specific shades of brown that occur in its object identification game. It may be the case, for example, that users are much more likely to mean the domain color `saddlebrownFigureObject` when they say *brown* than they are to mean `sandybrownFigureObject`. The hand-built model is blind to these kinds of regularities in the communicative intentions of COREF’s human dialogue partners.

The simplistic hand-built model was used because, as a system builder designing COREF, it was hard to know *a priori* what specific weights should be assigned to the different assumptions that factor into COREF’s construction of competing interpretations. In this section, we report on a proof-of-concept demonstration that these weights can be learned from the agent’s interactions with human users.

4.4.2.1 General approach

Our approach is based on the observation that COREF’s contribution tracking can be viewed as assigning a *fate* to every dialogue state it entertains as part of some thread of interpretation. In particular, if we consider the agent’s contribution tracking retrospectively, every dialogue state can be assigned a fate of *correct* or *incorrect*, where a state is viewed as correct if it or its descendants eventually capture all the probability mass that COREF is distributing across the viable surviving states, and incorrect otherwise. For example, looking retrospectively at the object subdialogue depicted in Figure 4.8 (page 154), the correct states are {s8756, s8923, s10488, s10747, s13025, s13141, s13146, s13211, s13275}, and the incorrect states are {s10507, s10757, s13274}. In this object subdialogue, the correct states capture the coherent thread of interpretation that COREF is able to converge on after asking its clarification question; the incorrect states s10507 and s10757 were temporarily entertained by COREF, but were eventually eliminated because they were determined not to represent the user’s intended contribution in saying *brown diamond*. The incorrect state s13274 is a state that COREF hypothesized in interpreting the user’s object-completing button click, but then immediately dropped.

In general, as this example shows, there are two ways that a state can end up with fate incorrect. One way is that the state and all of its descendants are eventually denied any probability mass due to a failure to interpret a subsequent utterance or action as a coherent contribution from any of those states. In this case, we say that the incorrect state was *eliminated*. (Eliminated states are indicated by a red ‘X’ in the contribution tracking figures in this dissertation.) The second way a state can end up incorrect is if COREF makes a strategic decision to drop the state, or all of its surviving descendants, at a time when the state or its descendants were assigned non-zero probability mass. In this case we say that the incorrect state was *dropped*. (Dropped states are indicated by a crosshatch pattern in the contribution tracking figures in this dissertation.)

We draw on this simple taxonomy of state fates to define an approach to learning the function $P(I = i_{t,j}|o, S_t = s_k)$ from data. To do so, we use the fact that COREF strategically drops down to the single most probable thread of interpretation at the moment each object is completed. This implies that, for each time t , there is a single hypothesized state at time t whose descendants will eventually (i.e. at some time $t + n$, where $n \geq 0$) capture all the probability mass that COREF assigns to viable states. In other words, for each time t , there is exactly one correct state s_k such that $S_t = s_k$, and further, COREF can retrospectively identify this state s_k as a side effect of its contribution tracking.

In estimating $P(I = i_{t,j}|o, S_t = s_k)$, then, we seek to estimate the probability that the correct interpretation of o is $i_{t,j}$ given that s_k is the correct state at time t . We define this as a learning problem by translating our taxonomy of state fates into a parallel taxonomy of interpretation fates. In particular, we say that an interpretation $i_{t,j}$ that is hypothesized under the assumption that $S_t = s_k$ is:

- correct* iff in fact, $S_t = s_k$, and states descendant from $i_{t,j}$ eventually capture all of COREF's probability mass;
- eliminated* iff in fact, $S_t = s_k$, and the last surviving state that is descendant from $i_{t,j}$ is eliminated;
- dropped* iff in fact, $S_t = s_k$, and the last surviving state that is descendant from $i_{t,j}$ is dropped;
- other* otherwise.

To illustrate these distinctions, we return to the clarification example of Figure 4.8 (page 154). In this example, the *correct* interpretations are $\{i_{1,1}, i_{2,1}, i_{3,1}, i_{4,1}, i_{5,1}, i_{6,1}, i_{7,1}, i_{8,1}\}$. The correct interpretations connect the correct states in a coherent thread of interpretation. The only *eliminated* interpretation is $i_{2,2}$. The only *dropped* interpretation is $i_{8,2}$. The only *other* interpretation is $i_{3,2}$. Note that the *other* status applies to interpretations which were hypothesized for states which were not correct.

Because we seek to estimate $P(I = i_{t,j}|o, S_t = s_k)$, which conditions the probability assigned to $I = i_{t,j}$ on the correctness of state s_k , we include in our training set only those of COREF's hypothesized interpretations which are *correct*, *eliminated*, or *dropped*. In particular, COREF's dialogue experience provides it with a large set of examples, each of the form $(\text{fate}(i_{t,j}), i_{t,j}, o, s_k)$ where $\text{fate}(i_{t,j}) \in \{\text{correct}, \text{eliminated}, \text{dropped}\}$ and where $i_{t,j}$ is the j^{th} interpretation COREF hypothesized for event o under the (ultimately correct) assumption that $S_t = s_k$.

To facilitate the use of a machine learning approach to estimating $P(I = i_{t,j}|o, S_t = s_k)$, we transform these examples as follows. We capture the fate of each interpretation $i_{t,j}$ in a discrete variable F whose value is *correct*, *eliminated*, or *dropped*. We also represent each intention $i_{t,j}$, observation o , and state s_k in terms of features. We summarize this transformation of our training examples as follows:

$$(\text{fate}(i_{t,j}), i_{t,j}, o, s_k) \longleftrightarrow (F = \text{fate}(i_{t,j}), \text{features}(i_{t,j}), \text{features}(o), \text{features}(s_k))$$

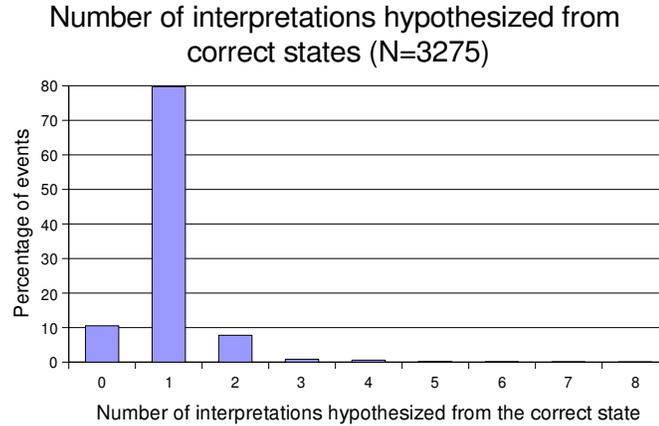


Figure 4.23: Distribution in number of interpretations hypothesized from correct states.

Under this transformation, we make the following identification:

$$P(I = i_{t,j} | o, S_t = s_k) = P(F = correct | features(i_{t,j}), features(o), features(s_k))$$

To summarize, we seek to learn the function $P(F = correct | features(i_{t,j}), features(o), features(s_k))$ from a set of training examples $E = \{e_1, \dots, e_n\}$ where, for $l = 1..n$, we have:

$$e_l = (F = fate(i_{t,j}), features(i_{t,j}), features(o), features(s_k)).$$

4.4.2.2 Training

We harvested a training set from COREF's contribution tracking for the 3275 sensory events that occurred in this user study. We began this process by assigning a fate to each state and interpretation that COREF entertained for each sensory event, as described in the previous section.

Of the 3275 sensory events that COREF interpreted, from the (retrospectively) correct state, the agent hypothesized 0 interpretations for 345 events, 1 interpretation for 2612 events, and more than one interpretation for 318 events. The overall distribution in the number of interpretations hypothesized from the correct state is given in Figure 4.23.

For this proof-of-concept implementation, we chose to train a maximum entropy model (Berger et al., 1996) to estimate $P(F = correct | features(i_{t,j}), features(o), features(s_k))$. We used the MALLET maximum entropy classifier (McCallum, 2002) as an off-the-shelf, trainable maximum entropy model. For features, we defined a range of potentially useful features, which we list in Figures 4.24,

feature name	feature description
NumTacitActions	The number of tacit actions in $i_{t,j}$.
TaskActionsTagger	Generates features describing each action a_k in $i_{t,j} = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$.
ActorDoesTaskActionTagger	Generates features describing each $A_k : a_k$ in $i_{t,j} = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$.
PresuppositionsTagger	If o is an utterance, generates features describing each presupposition assigned to o by $i_{t,j}$.
AssertionsTagger	If o is an utterance, generates features describing each dialogue act assigned to o by $i_{t,j}$.
SyntaxTagger	If o is an utterance, generates one feature describing the syntactic analysis assigned to o by $i_{t,j}$.
FlexiTaskIntentionActorsTagger	Given $i_{t,j} = \langle A_1 : a_1, A_2 : a_2, \dots, A_n : a_n \rangle$, generates one feature describing the actor sequence $\langle A_1, A_2, \dots, A_n \rangle$ in $i_{t,j}$.

Figure 4.24: The interpretation features, $\text{features}(i_{t,j})$, in our learned model.

4.25, and 4.26. These features capture a variety of pragmatic details that seemed as though they might provide heuristic clues to the correct interpretation for a user utterance or action. To allow these various kinds of features (integer-valued, binary-valued, and string-valued) to interface to the maximum entropy model, these features were converted into a much broader class of indicator features taking on a value of either 0.0 or 1.0.

We are generally interested in whether COREF's experience with previous subjects can be lever-

feature name	feature description
WordsTagger	If o is an utterance, generates features that indicate the presence of each word that occurs in the utterance.

Figure 4.25: The observation features, $\text{features}(o)$, in our learned model.

feature name	feature description
NumTasksUnderway	The number of tasks underway in s_k .
TasksUnderwayTagger	The name, stack depth, and graph task state for each task underway in s_k .
NumRemainingReferents	The number of objects yet to be identified in s_k .
TabulatedFactsTagger	Generates features describing each proposition in the conversational record in s_k .
CurrentTargetConstraintsTagger	Generates features describing each positive and negative constraint on the current target in s_k .
UsefulPropertiesTagger	Generates features describing each property instantiated in the experiment interface in s_k .

Figure 4.26: The dialogue state features, $\text{features}(s_k)$, in our learned model.

aged to improve its interactions with new subjects. Therefore, to evaluate our approach, while making maximal use of our available data set, we performed a hold-one-subject-out cross-validation using our 20 human subjects $H = \{h_1, \dots, h_{20}\}$. That is, for each subject h_i , we trained a model on the training examples associated with subjects $H \setminus \{h_i\}$, and then tested the model on the examples associated with subject h_i . Formally, for each subject $h_i \in H$, we define $\text{TRAINING-DATA}(h_i)$ to be the set of training examples harvested from COREF’s interactions with subjects $H \setminus \{h_i\}$, and $\text{TEST-DATA}(h_i)$ to be the set of examples harvested from COREF’s interaction with subject h_i . Our procedure, then, is to train a maximum entropy model of

$$P(F = \text{correct} \mid \text{features}(i_{t,j}), \text{features}(o), \text{features}(s_k))$$

on $\text{TRAINING-DATA}(h_i)$, and test that model on $\text{TEST-DATA}(h_i)$, for each $h_i \in H$.

Training for each subject h_i consisted of applying two steps to $\text{TRAINING-DATA}(h_i)$. In the first step, specific features were selected using MALLETT’s feature selection algorithm, which incrementally selects features (as well as conjunctions of features) that maximize an exponential gain function which represents the value of the feature in predicting interpretation fates. Based on manual experimentation, we chose to have MALLETT select about 300 features for each learned model. In the second step, the selected features were used to train the model to estimate $P(F =$

$correct \mid \text{features}(i_{t,j}), \text{features}(o), \text{features}(s_k)$). We used MALLET’s implementation of Limited-Memory BFGS (Nocedal, 1980) to train the model.

4.4.2.3 Results

The result of training is 20 separate maximum entropy models, one for each subject. To complete our cross-validation, we used each model to assign a numeric probability,

$$P(F = correct \mid \text{features}(i_{t,j}), \text{features}(o), \text{features}(s_k)),$$

to each training example $e_l = (F = \text{fate}(i_{t,j}), \text{features}(i_{t,j}), \text{features}(o), \text{features}(s_k))$ in the test set for that model.

It is difficult to evaluate these numeric probabilities directly, however. Further, we are chiefly interested in how these models perform when they face an ambiguous interpretation for a sensory event. We therefore limit our analysis to the 318 events for which multiple interpretations were hypothesized in the correct state in the user study. To assess our results, we investigate the way the probabilities assigned by the learned models rank the competing interpretations. In particular, we are interested in the frequency with which the interpretation that is known to be *correct* for a test event is assigned higher probability, and thus ranked higher, than the competing interpretations.

We present these results in Figures 4.27 and 4.28. These figures show the relative frequency with which the correct interpretation is assigned specific ranks by the learned and hand-built models. The figures show that one outcome that can occur, especially with the hand-built model, is that competing interpretations are assigned exactly the same probability. In this case, the correct interpretation cannot be distinguished from one or more alternative interpretations by probability. As can be seen in equation (4.1) on page 194, the hand-built model will assign equal probability to any two interpretations that include the same number of tacit actions. This is the case, for example, for the two interpretations $i_{2,1}$ and $i_{2,2}$ which COREF assigns to the user’s utterance *brown diamond* in Figure 4.8 (page 154). The learned model can use additional features to make finer-grained distinctions between competing interpretations.

Figures 4.27 and 4.28 generally show that the learned model does a better job of assigning higher probability to correct interpretations than the hand-built model does. For example, for 226 (88.6%) of the 255 2-way ambiguities that COREF encountered in this user study, the learned model — which has in every case been trained on COREF’s experience talking to other users — assigned higher probability to the interpretation which COREF ultimately judged to be correct

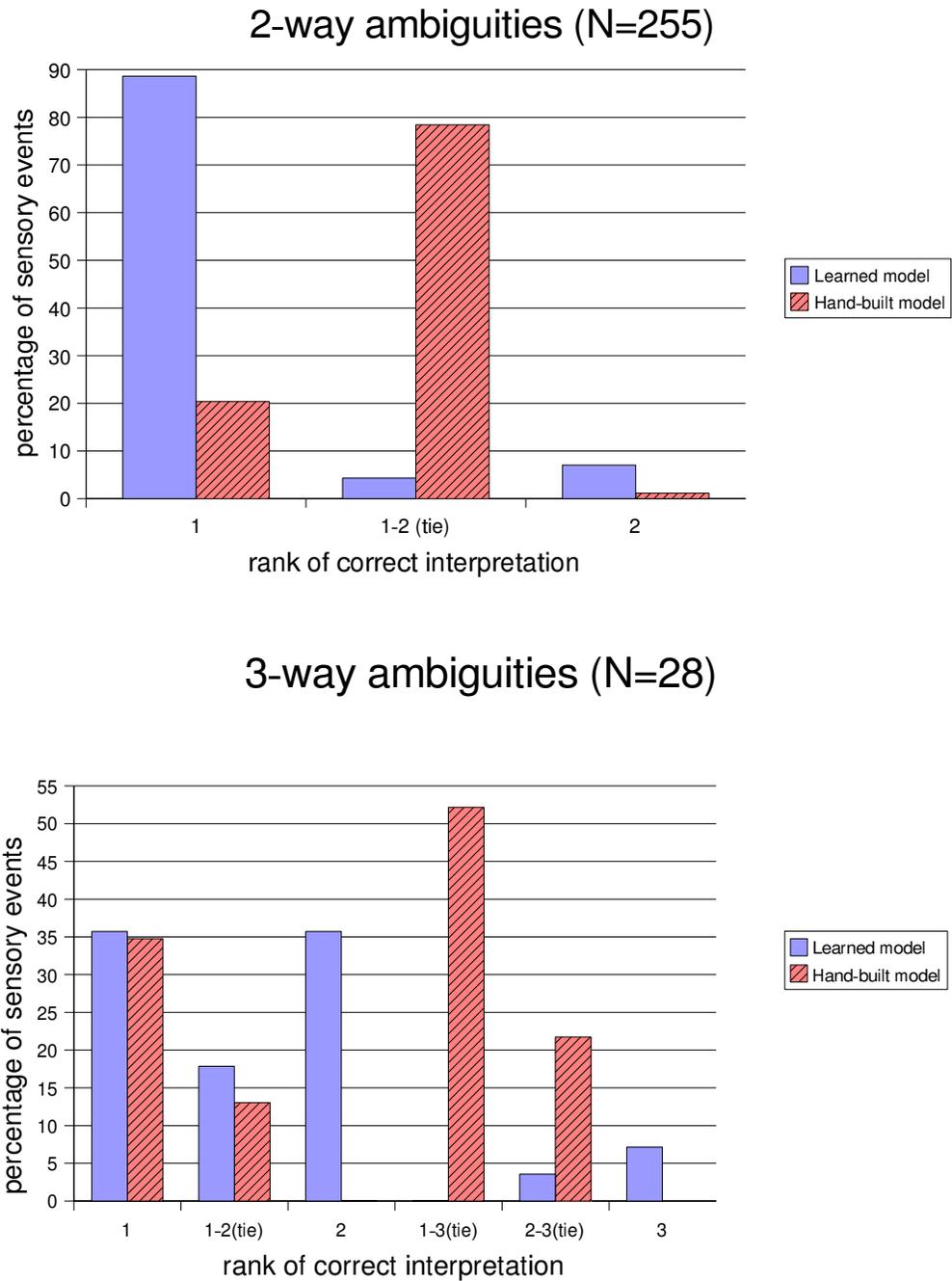


Figure 4.27: Comparison of learned and hand-built models for 2- and 3-way ambiguities.

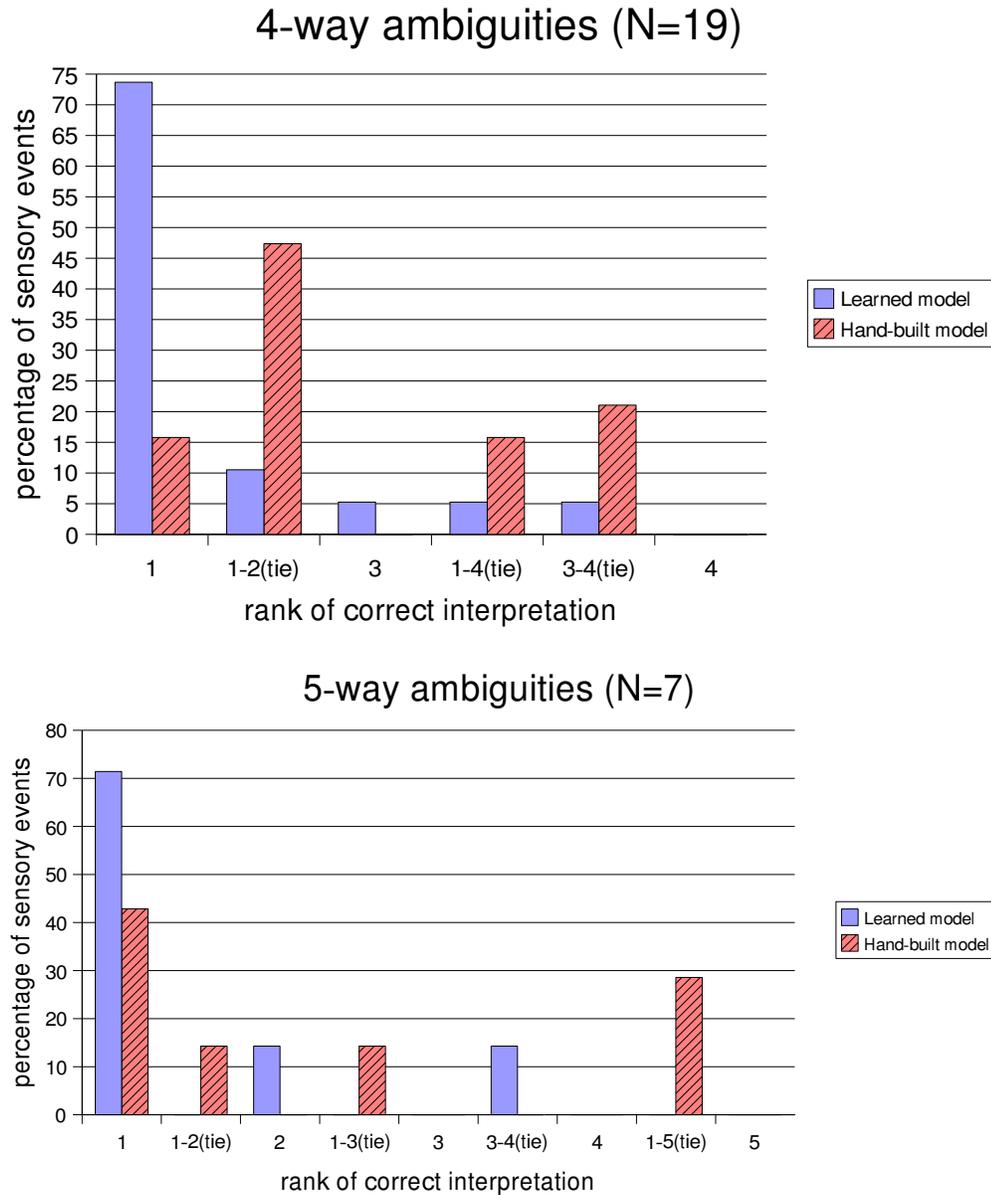


Figure 4.28: Comparison of learned and hand-built models for 4- and 5-way ambiguities.

than it did to the competing interpretation, which COREF ultimately rejected. This is a *prima facie* indication that the learned model does a better job of distributing probability mass onto correct interpretations. The results are similar for the 4-way and 5-way ambiguities, where the learned model clearly outperforms the hand-built model at assigning higher probability to correct interpretations than to incorrect interpretations. In the case of 3-way ambiguities, the results are less clear cut, but the learned model does seem preferable to the hand-built model. In particular, the learned model breaks many ties left by the hand-built model, and does so in ways that tend to rank the correct interpretation first or second among the three interpretations.

Taken together, these results show that COREF could make productive use of its dialogue experience by learning to assign probability based on features it has historically associated with correct utterance interpretations. We present these results as a proof-of-concept that contribution tracking provides a source of information that an agent can use to improve its statistical interpretation process, rather than as a definitive statement of the best way to go about doing this, or of the results that could be expected in other dialogue systems. Our work to date on this kind of data-driven interpretation has a number of limitations. First, although 318 ambiguous interpretations did occur, this user study provided a relatively small number of ambiguous interpretations, in machine learning terms; most (80.2%) of those that did occur were 2-way ambiguities. We discuss limitations in the kinds of ambiguities COREF encounters further in Chapter 5.

Second, this learning experiment has been performed after the fact, and we have not yet investigated the performance of the learned model in a follow-up experiment in which COREF uses the learned model in interactions with its users. Such a further experiment could demonstrate the usefulness of the learned model, for example in terms of a reduction in the number of clarification questions that are necessary, or a reduction in the number of *wrong object* outcomes COREF encounters.

A third limitation lies in the definition of the ‘correct’ fate for interpretations, which to some extent conflates the user’s actual intentions with COREF’s subsequent assumptions about the user’s intentions. Specifically, if COREF decides to strategically drop the user’s *actual intended interpretation*, this may lead another interpretation to be marked as ‘correct’ in the scheme we have implemented here. While we believe this heuristic approach has clear value in exploiting the agent’s ability to resolve ambiguities, whatever that ability may be, there is plenty of room for further investigation of alternative approaches to harvesting meaningful examples of correct and incorrect interpretations from an agent’s dialogue experience.

The idea of using a language-using agent’s experience to improve its interpretation models is not

new. There are several related approaches in the literature. For example, Bohus et al. (2008) use users' confirmations of their spoken requests in a multi-modal interface to improve the system's interpretation of subsequent utterances. Many AI researchers have also attempted to learn vocabulary items from their users by connecting user vocabulary to the agent's perceptual representations at the time of utterance (Oates et al., 2000; Roy and Pentland, 2002; Cohen et al., 2002; Yu and Ballard, 2004; Steels and Belpaeme, 2005; DeVault et al., 2006). The contribution of this particular learning experiment is to show that we can leverage the ability of a collaborative dialogue agent to flexibly resolve its own uncertainty over time to improve the agent's interpretation models. This perspective suggests that a focus on improving our agents' basic abilities to tolerate and resolve ambiguities as a dialogue proceeds — rather than a focus on building agents that can recognize intentions accurately out of the box — may prove to be a valuable technique for improving the overall dialogue competence of the agents we build.

4.5 Conclusion

In this chapter, we have presented an empirical evaluation of the COREF agent. We have shown that COREF is often able to complete its object identification subdialogues under a single thread of interpretation. When the agent does encounter ambiguity, it has a limited ability to proceed despite its uncertainty and complete its object identification subdialogues successfully. We have further shown that COREF's contribution tracking allows it to capture and reason about ambiguities in user acknowledgments, and to approach the management of these and other ambiguities as a collaborative endeavor in which it continues to attempt to make recognizable contributions, and in which the agent's uncertainty can be resolved flexibly as new information becomes available. Finally, we have quantified COREF's ability to reduce its own uncertainty in specific situations, and presented a proof-of-concept demonstration that COREF's experience in managing its own uncertainty can be used to improve its probabilistic interpretation model.

In the next chapter, we conclude the dissertation and discuss some of the limitations of the work we have done to date.

Chapter 5

Conclusion and limitations

We now conclude the dissertation with a summary and a discussion of some limitations and future research directions suggested by this work.

5.1 Summary

In this dissertation, we have presented *contribution tracking* as a pragmatic reasoning process by which implemented dialogue agents can organize and manage their uncertainty as their conversations evolve. This process allows agents to:

1. spawn multiple threads of interpretation in response to perceived ambiguities;
2. flexibly redistribute probability mass among the different threads of interpretation as the dialogue unfolds; and
3. carry out principled, context-sensitive interpretation and generation of utterances in a way that:
 - (a) maintains coherence within the individual threads of interpretation, and
 - (b) allows the agent to make recognizable contributions despite its uncertainty.

Taken together, these features substantially clarify the theoretical and methodological target of collaborative language use in implemented dialogue systems.

To present this vision, we began in Chapter 2 by defining *contributive action* and *contributive language use* as public attempts to advance a two-agent, task-oriented activity or dialogue in a recognizable way. In Chapter 3, we built on these definitions to define and present *contribution tracking*. We introduced RUBRIC as a modular computational architecture within which contribution tracking may be implemented. We presented the COREF agent as an implemented instance of this architecture, and as a demonstration of how contribution tracking can be used to organize a collaborative agent's linguistic reasoning when it faces uncertainty. In Chapter 4, we evaluated

COREF’s use of contribution tracking in a user study. Detailed examples of COREF’s pragmatic reasoning in the observed dialogues served to illustrate how COREF uses contribution tracking to try to act and speak collaboratively in its object identification game. We also used the study to articulate several new qualitative and quantitative capabilities for implemented dialogue systems. These abilities include representing and reasoning about ambiguous acknowledgments, treating ambiguity management as a flexible collaborative activity, quantifying an agent’s ability to act to reduce its own uncertainty using general purpose collaborative reasoning, and using an agent’s own experience to tune its probabilistic interpretation models.

More broadly, contribution tracking suggests that the sociological and methodological divisions between two currently prominent approaches to dialogue system building may be more reconcilable than they appear to be. In work that aims to optimize global task performance (Section 2.3.1, page 16), it has been common to employ probabilistic models of dialogue state, and to give relatively little explicit attention to traditional linguistic analyses of utterances or to the numerous context-sensitive aspects of human utterance interpretation. Conversely, in work that has aimed to replicate the fine-grained local coherence that context-based linguistic coordination creates in human-human dialogue, there has been a tendency to employ heuristically updated, unitary representations of common ground (Section 2.3.2.3, page 23) rather than probabilistic models of dialogue state. Contribution tracking suggests that dialogue systems researchers can draw from the strengths of both approaches. In particular, it shows that probabilistic inference can be a part of the pragmatic reasoning that uncertain speakers use to achieve fine-grained, context-based linguistic coordination in their dialogues. By clarifying this reasoning process, contribution tracking allows us to continue to view common ground as an eventual outcome of many dialogues and subdialogues, while expanding the practical routes by which a collaborative speaker can try to achieve this outcome over time.

5.2 Limitations and future work

The work presented here has a number of limitations. In this section, we discuss some of the more salient limitations and sketch some future research directions that might start to address them.

5.2.1 Other kinds of uncertainty

One problem that many spoken dialogue systems face is the high error rates in automatic speech recognition results; see e.g. Vargès and Purver (2006). The work reported in this dissertation sidesteps this important problem through the use of a teletype interface for COREF’s dialogues.

To date, we have not investigated the utility of contribution tracking as a strategy for managing uncertainty arising from automatic speech recognition results.

In principle, we believe contribution tracking could be applied to perceived ambiguities in the words users have uttered to an agent, or more generally, to perceived ambiguities in phonology and syntax. A theoretical starting point here would be the work of Ginzberg and Cooper (2004). We might proceed by trying to translate perceived phonological or syntactic ambiguities into ambiguities in the context resulting from an utterance, analogously to COREF’s current response to other kinds of ambiguities. However, if real-world speech recognition hypotheses were employed (e.g. by inspecting the probabilistic word lattice constructed by the recognizer), it is likely that more compact representations of the agent’s uncertainty about context, and more efficient context-sensitive interpretation algorithms, would be required. For some existing work along these lines, see Schuler et al. (In Press) or Wu et al. (2008).

The issue of needing more compact representations of the agent’s uncertainty is more general. COREF’s current limitation of tracking a maximum of 3 threads of interpretation reflects the linear relationship between the number of threads considered and COREF’s run-time interpretation and generation speed. In applications where speaker meanings tie into continuous domain models, or where the agent would need to entertain more wide-ranging discrete hypotheses about the user’s intended contribution, it would likely be necessary to develop more compact context representations and more aggressive ambiguity management strategies than those used by COREF.

5.2.2 Other clarification and grounding phenomena

Contribution tracking is most directly applicable to situations in which the speaker’s intended contribution is among the alternative interpretations hypothesized by the hearer. This is the case for the ambiguous acknowledgments and other ambiguous speaker meanings that occur frequently in COREF’s domain, and that have been our focus to date. While contribution tracking allows COREF to ask and answer questions to manage these ambiguities, contribution tracking does not immediately cover clarification questions that are not designed to resolve perceived ambiguities, but rather are asked in situations where no interpretations are found. Such examples occur; see Ginzberg and Cooper (2004) or Purver (2004) for examples. As discussed in Section 3.3.8.1 (page 119), when COREF finds no interpretations for a user utterance, it notes the occurrence of the utterance and signals an interpretation failure (currently by saying *umm*), but it otherwise leaves its threads of interpretation mostly as they were, and is unable to address the failure with its usual ambiguity man-

agement policy. More detailed characterizations of agents’ reasoning in such cases are still required, and work such as Purver’s provides a natural starting point.

Moreover, traditional classifications of grounding actions (Traum, 1999b) include a variety of other cases as well. For example, we have not modeled repair requests like *what?* or *what did you say?*, which can signal interpretation failure or the hearer’s incredulity at the speaker’s apparent (but correctly and uniquely identified) meaning. Similarly, we do not treat self-repairs by speakers. These can exclude a possible but unintended interpretation, to avoid a foreseen misunderstanding—an example in COREF’s domain would be, *A: I moved it. A: I mean I moved the blue circle.* Self-repairs can also correct a prior verbal mistake, as when a speaker has mistakenly used the wrong word: *A: I moved the circle. A: I mean I moved the square.* It would be interesting to explore whether richer models of interlocutor uncertainty and more extensive models of grounding-related dialogue acts (Traum, 1994) would enable us to account for these utterance types in a contribution tracking framework.

5.2.3 Reducing the use of hand-crafted domain-specific models

COREF relies on a number of hand-crafted and somewhat domain-specific models, including its grammar, its task and action models, and its distinction between possible and acceptable actions in interpretation and generation. These models allow COREF to construct relatively detailed models of the contributions utterances can make in context, but building such models constitutes a substantial effort in terms of the time and various kinds of theoretical and technical expertise that are required. We believe that the effort required to build these kinds of models can be substantially reduced through data-based techniques. Example-based generation techniques (Wong and Mooney, 2007; Stone, 2003; Varges and Mellish, 2001) can potentially help reduce the authoring burden of building up a COREF-style grammar for dialogue; our own efforts in this direction are presented in DeVault et al. (2008a) and DeVault et al. (2008b). Bangalore et al. (2006) explore the acquisition of hierarchical dialogue task models, which are somewhat similar to COREF’s, from annotated dialogue data. We discuss the optimization of dialogue policy — which would effectively automate COREF’s distinction between acceptable and possible actions — in the next section.

5.2.4 Optimizing agent performance

This dissertation suggests that collaborative dialogue agents face uncertainty from various sources, but that their experience can provide quantitative evidence about what kinds of uncertainty arise and

how best to resolve them. Some recent POMDP-based approaches to dialogue policy optimization (e.g. Williams and Young, 2006, 2007; Rieser and Lemon, 2008) try to use a dialogue agent's experience (or simulations of dialogue experience) to frame dialogue policy selection as a reinforcement learning problem. It would be interesting to investigate whether a contribution tracking agent could employ a similar policy optimization approach to optimize trade-offs between asking clarification questions, assuming the most likely interpretation, or simply proceeding with an uncertain context.

5.2.5 Learning to speak collaboratively

Finally, the machine learning experiment described in Section 4.4.2 (page 193) has only scratched the surface of the various ways a contribution tracking dialogue agent could use its interpretive experience to learn to make linguistic choices that its users will find more natural and understandable in context. This research direction raises a number of complex and fascinating issues, including how we can quantify a speaker's fine-grained linguistic choices as situated among a spectrum of relatively collaborative choices it could have made, as well as the possibility that a dialogue agent could draw on the existing linguistic and collaborative skills of its dialogue partners to help it connect new vocabulary and concepts to its broader perceptual and social environment (DeVault et al., 2006). That we are approaching the point when we can begin to give precise computational answers to these kinds of questions makes the next few decades in dialogue systems research seem very exciting indeed.

Appendix A

COREF task instructions

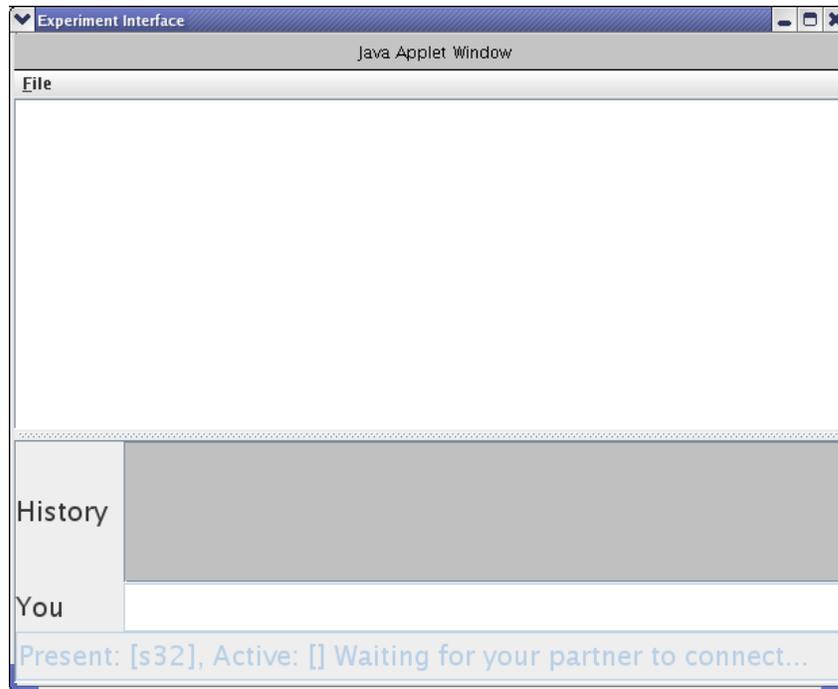
Figures A.1, A.2, and A.3 present the exact instructions that participants in our COREF experiments were given.

Task Instructions

In this experiment, your task will be to create a visual scene with a computer program acting as your partner. This computer program is an interactive dialogue program, which means it can talk back and forth with you while you work on the task together. In these instructions, we call this computer program an "agent" and refer to it as "your partner". The agent cannot see or hear you, and it cannot see what is on your computer monitor. If you like, you may think of the agent as if it were a person located in a separate room, working on a separate computer that is similar to yours. We will sometimes speak as though the agent were a "person in another room", looking at their own computer monitor, in these instructions.

Starting the Experiment Interface

After you have read through these instructions, you will be directed to start the computer software that will connect you with your partner and allow you to complete the task. We call this software the "Experiment Interface". When you start the Experiment Interface, a window will appear that looks like this:



When you first start the Experiment Interface, your partner may not be available yet. Thus, you may see the message "Waiting for your partner to connect...", as illustrated in the above screenshot. If you see this message, please simply wait a few moments. Once your partner is connected, the message will disappear and the experiment will continue.

Figure A.1: COREF task instructions: page 1 of 3.

Talking to your partner

Throughout the experiment, you and your partner may talk back and forth using written English. You may type anything you want to say into the text field labeled "You" near the bottom of the window. (Sometimes you might need to click on the text field with the mouse, in order to move the text cursor there, before you can type.) When you press the [Enter] key, anything you have typed into the text field will be transmitted to your partner. What you have said will also appear, along with anything your partner has said, in the part of the screen labeled "History".

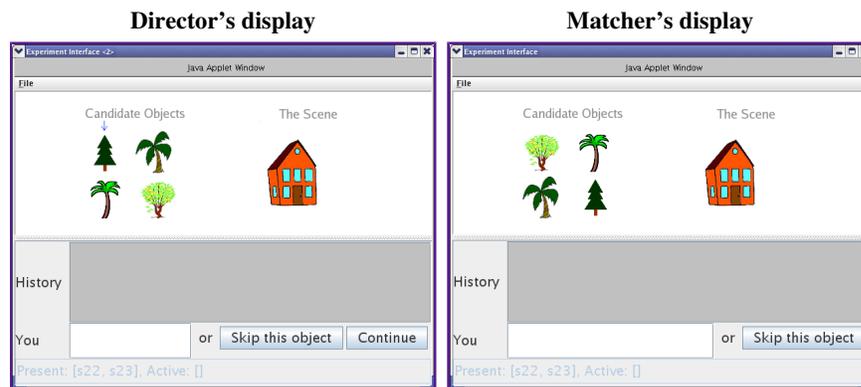
During this experiment, please do not attempt to speak out loud to your partner, or otherwise communicate with your partner outside of the Experiment Interface.

Your task: create a visual scene

In this experiment you will work with your partner to create a visual scene. To create the scene, each of you will select particular objects to add into your own private version of the scene. Your private version of the scene will be displayed in the Experiment Interface. Your goal is to add the same objects to *your* private scene that your partner adds to *their* scene. This way, at the end of the task, you and your partner will have created the exact same scene.

To help you coordinate your choices, one of you will play the role of Director, and the other will play the role of Matcher. (The Experiment Interface will tell you which role you will play.)

The Director and the Matcher will add new objects into their private scenes one by one. In selecting each new object, both the Director and the Matcher will see a set of candidate objects that could be added to the scene. However, the candidate objects will appear in a shuffled order on their individual screens. For example, the Director and Matcher might see:



Important note: Observe that while the Director and Matcher see the same candidate objects, the candidate objects appear in shuffled positions on their screens. For example, the object located at the top left of the Director's display above is not located at the top left of the Matcher's display.

Once the candidate objects are displayed, the Experiment Interface will reveal to the Director, *but not to the Matcher*, which one of the objects should be added to the visual scene. This "target" object will be visually highlighted by an arrow appearing in the Director's private

Figure A.2: COREF task instructions: page 2 of 3.

display, as illustrated above.

The goal is then for the Director to convey to the Matcher which one of the candidate objects is the target object. The Director and Matcher may talk back and forth as much as they want to achieve this.

If you are the Matcher, once you realize which object is the target, click on it once and it will be added into to your visual scene. If you realize you have added the wrong object to the scene, you may click on the object again to remove it, and then add another object. The task proceeds to the next object in the scene when the Director clicks the "Continue" button. Once the Director clicks this button, the target object is added to the Director's private scene, and the Matcher's choice becomes irreversible.

In this experiment, it is very important to be accurate, but you should move right long, as you are being timed.

If for any reason you feel like you are no longer making progress on a particular object, you always have the option to skip that object by clicking the "Skip this object" button. Please do not skip an object just because it is taking a little longer than you would like. You should only skip an object if you feel you are no longer making progress with your partner.

Your interaction with your partner as you complete the task will be digitally recorded.

If you wish, you may read through these instructions again. Take your time and be sure you understand the task.

When you are ready, press "Continue" to start the Experiment Interface.



Continue

Document version: \$Revision: 1.3 \$ \$Date: 2006/03/23 17:44:49 \$

Figure A.3: COREF task instructions: page 3 of 3.

Bibliography

- Allen, Byron, Dzikovska, Ferguson, Galescu, and Stent. An architecture for a generic dialogue shell. *Journal of Natural Language Engineering*, 6(3):1–16, 2000.
- J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. Towards conversational human-computer interaction. *AI Magazine*, 22(4):27–37, 2001a.
- James F. Allen and C. Raymond Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15(3):143–178, 1980.
- James F. Allen, George Ferguson, and Amanda Stent. An architecture for more realistic conversational systems. In *Intelligent User Interfaces*, pages 1–8, 2001b.
- J. L. Austin. *How to do things with words*. Oxford University Press, 1962.
- Srinivas Bangalore, Giuseppe Di Fabbrizio, and Amanda Stent. Learning the structure of task-driven human-human dialogs. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 201–208. Association for Computational Linguistics, 2006.
- C. Barker. The dynamics of vagueness. *Linguistics and Philosophy*, 25(1):1–36, 2002.
- David Beaver. *Presupposition and Assertion in Dynamic Semantics*. CSLI Publications, Stanford, California, 2001. ISBN 1575861208 (pbk), 1575861208 (cloth).
- Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.
- Nathan James Blaylock. *Towards Tractable Agent-based Dialogue*. Ph.D. dissertation, Department of Computer Science, University of Rochester, Rochester, New York, 2005.
- D. Bohus and A. Rudnicky. A k hypotheses + other belief updating model. In *Proceedings AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, 2006.
- Dan Bohus, Xiao Li, Patrick Nguyen, and Geoffrey Zweig. Learning n-best correction models from implicit user feedback in a multi-modal local search application. In *The 9th SIGdial Workshop on Discourse and Dialogue*, 2008.
- Michael E. Bratman. Shared cooperative activity. *The Philosophical Review*, 101(2):327–341, 1992.
- Susan E. Brennan and Herbert H. Clark. Conceptual pacts and lexical choice in conversation. *Journal of Experimental Psychology*, 22(6):1482–1493, 1996.
- Harry Bunt. Dialogue pragmatics and context specification. In Harry Bunt and William Black, editors, *Abduction, Belief and Context in Dialogue. Studies in Computational Pragmatics*, pages 81–150. Amsterdam: Benjamins, 2000.
- Harry Bunt. Interaction management functions and context representation requirements. In S. LuperFoy, A. Nijholt, and G. Veldhuizen van Zanten, editors, *Dialogue Management in Natural Language Systems. Proc. of 11th Twente Workshop on Language Technology, University of Twente, Enschede*, pages 187–198. 1996.

- Harry C. Bunt. Context and dialogue control. *THINK Quarterly*, 3:19–31, 1994. URL cite-seer.ist.psu.edu/bunt94context.html.
- Donna K. Byron. Resolving pronominal reference to abstract entities. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 80–87, 2002.
- Malinda Carpenter, Katherine Nagell, and Michael Tomasello. Social cognition, joint attention, and communicative competence from 9 to 15 months of age. *Monographs of the Society for Research in Child Development*, 63(4, Serial No. 255), 1998.
- Jennifer Chu-Carroll and Sandra Carberry. A plan-based model for response generation in collaborative task-oriented dialogues. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI)*, pages 799–805, 1994. URL citeseer.ist.psu.edu/chu-carroll194planbased.html.
- H. H. Clark. *Arenas of Language Use*. University of Chicago, 1992.
- H. H. Clark and E. F. Schaefer. Contributing to discourse. *Cognitive Science*, 13:259–294, 1989.
- Herbert Clark. *Using Language*. Cambridge University Press, Cambridge, UK, 1996.
- Herbert H. Clark and Catherine R. Marshall. Definite reference and mutual knowledge. In Arivind Joshi, Bonnie Webber, and Ivan Sag, editors, *Elements of Discourse Understanding*, pages 10–63. Cambridge University Press, Cambridge, England, 1981.
- H.H. Clark and D. Wilkes-Gibbs. Referring as a collaborative process. In Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, pages 463–493. MIT Press, Cambridge, Massachusetts, 1990, 1986.
- Paul R. Cohen, Tim Oates, Carole R. Beal, and Niall Adams. Contentful mental states for robot baby. In *Eighteenth national conference on Artificial intelligence*, pages 126–131, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence. ISBN 0-262-51129-0.
- Phil Cohen. Dialogue modeling. In Ronald Cole, Joseph Mariani, Hans Uszkoreit, Giovanni Battista Varile, Annie Zaenen, and Antonio Zampolli, editors, *Survey of the State of the Art in Human Language Technology (Studies in Natural Language Processing)*, pages 204–210. Cambridge University Press, 1997.
- Philip R. Cohen and Hector J. Levesque. Performatives in a rationally based speech act theory. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 79–88, 1990a. URL citeseer.ist.psu.edu/article/cohen90performatives.html.
- Philip R. Cohen and Hector J. Levesque. Rational interaction as the basis for communication. In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 221–255. MIT Press, Cambridge, Massachusetts, 1990b.
- Philip R. Cohen and Hector J. Levesque. Teamwork. *Nous*, 25:11–24, 1991.
- Mark G. Core and James F. Allen. Coding dialogues with the DAMSL annotation scheme. In David Traum, editor, *Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35, Menlo Park, California, 1997. American Association for Artificial Intelligence. URL citeseer.ist.psu.edu/core97coding.html.
- Robert Dale and Ehud Reiter. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263, 1995.
- David DeVault and Matthew Stone. Scorekeeping in an uncertain language game. In *Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue (SemDial-10)*, pages 139–146, 2006.

- David DeVault and Matthew Stone. Managing ambiguities across utterances in dialogue. In *Proceedings of the 11th Workshop on the Semantics and Pragmatics of Dialogue (Decalog 2007)*, pages 49–56, 2007.
- David DeVault, Charles Rich, and Candace L. Sidner. Natural language generation and discourse context: Computing distractor sets from the focus stack. In *Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2004)*, pages 887–892, 2004.
- David DeVault, Natalia Kariaeva, Anubha Kothari, Iris Oved, and Matthew Stone. An information-state approach to collaborative reference. In *ACL 2005 Proceedings Companion Volume. Interactive Poster and Demonstration Sessions*, pages 1–4, University of Michigan, 2005.
- David DeVault, Iris Oved, and Matthew Stone. Societal grounding is essential to meaningful language use. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, pages 747–754, 2006.
- David DeVault, David Traum, and Ron Artstein. Practical grammar-based NLG from examples. In *Fifth International Natural Language Generation Conference (INLG)*, 2008a.
- David DeVault, David Traum, and Ron Artstein. Making grammar-based generation easier to deploy in dialogue systems. In *Ninth SIGdial Workshop on Discourse and Dialogue (SIGdial)*, 2008b.
- Philip Edmonds. A computational model of collaboration on reference in direction-giving dialogues. Msc thesis, also published as technical report csri-289, Department of Computer Science, University of Toronto, 1993.
- Raquel Fernández. *Non-Sentential Utterances in Dialogue: Classification, Resolution and Use*. PhD thesis, Department of Computer Science, King’s College London, University of London, UK., 2006.
- R. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.
- György Gergely. The development of understanding self and agency. In U. Goshwami, editor, *Blackwell Handbook of Childhood Cognitive Development*, pages 26–46. Oxford: Blackwell, 2002.
- György Gergely, Zoltán Nádasy, Gergely Csibra, and Szilvia Bíró. Taking the intentional stance at 12 months of age. *Cognition*, 56:165–193, 1995.
- Jonathan Ginzberg and Robin Cooper. Clarification, ellipsis and the nature of contextual updates in dialogue. *Linguistics and Philosophy*, 27(3):297–365, 2004.
- Alvin Goldman. *A Theory of Human Action*. Prentice-Hall, 1970.
- Paul Grice. Meaning. *Philosophical Review*, 66(3):377–388, 1957.
- Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.
- Barbara J. Grosz and Candace L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.
- Barbara J. Grosz and Candace L. Sidner. Plans for discourse. In Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, pages 417–444. MIT, 1990.
- Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. In *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, pages 50–61, 1984. URL citeseer.ist.psu.edu/halpern84knowledge.html.
- Gerlind Hauser, Tanya Behne, Malinda Carpenter, and Michael Tomasello. How children understand misunderstandings of their requests. unpublished.

- Peter A. Heeman and Graeme Hirst. Collaborating on referring expressions. *Computational Linguistics*, 21(3):351–383, 1995.
- Irene Heim. *The Semantics of Definite and Indefinite Noun Phrases*. PhD thesis, Linguistics Department, University of Massachusetts, University of Massachusetts, Amherst, 1982.
- J. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as abduction. *Artificial Intelligence*, 63:69–142, 1993.
- Eric Horvitz and Tim Paek. Harnessing models of users’ goals to mediate clarification dialog in spoken language systems. In *Proceedings of the Eighth International Conference on User Modeling*, pages 3–13, 2001.
- Nicholas R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75(2):195–240, 1995.
- Hans Kamp. A theory of truth and semantic representation. In Jeroen A. Groenendijk, Theo Janssen, and Martin Stokhof, editors, *Formal Methods in the Study of Language*. Foris, Dordrecht, 1981.
- Hans Kamp and Uwe Reyle. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht, 1993.
- S. Larsson and D. Traum. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, 6:323–340, 2000.
- Hector J. Levesque, Philip R. Cohen, and José H. T. Nunes. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI)*, pages 94–99, 1990.
- E. Levin, R. Pieraccini, and W. Eckert. Using markov decision process for learning dialogue strategies. In *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1998. URL citeseer.ist.psu.edu/article/levin98using.html.
- Esther Levin and Roberto Pieraccini. A stochastic model of computer-human interaction for learning dialogue strategies. In *Proceedings of Eurospeech*, pages 1883–1886, Rhodes, Greece, 1997. URL citeseer.ist.psu.edu/levin97stochastic.html.
- David K. Lewis. *Convention: A Philosophical Study*. Harvard University Press, Cambridge, Massachusetts, 1969.
- Ulf Liszkowski, Malinda Carpenter, and Michael Tomasello. Reference and attitude in infant pointing. *Journal of Child Language*, 34:1–20, 2007.
- Karen E. Lochbaum. A collaborative planning model of intentional structure. *Computational Linguistics*, 24(4):525–572, 1998.
- C. Matheson, M. Poesio, and D. Traum. Modelling grounding and discourse obligations using update rules. In *Proceedings of NAACL*, May 2000. URL citeseer.ist.psu.edu/article/matheson00modelling.html.
- Andrew McCallum. MALLETT: A MACHINE learning for LanguagE toolkit, 2002. <http://mallet.cs.umass.edu>.
- Dov Monderer and Dov Samet. Approximating common knowledge with common beliefs. *Games and Economic Behavior*, 1(2):170–190, 1989.
- Roser Morante, Simon Keizer, and Harry Bunt. A dialogue act based model for context updating. In *Proceedings of the 11th Workshop on the Semantics and Pragmatics of Dialogue (Decalog 2007)*, pages 9–16, 2007.

- Ranjit Nair and Milind Tambe. Hybrid BDI-POMDP framework for multiagent teaming. *Journal of Artificial Intelligence (JAIR)*, 23:367–420, 2005.
- Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, 1980.
- David G. Novick and Stephen Sutton. An empirical model of acknowledgment for spoken-language systems. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 96 – 101, New Mexico State University, Las Cruces, New Mexico, 1994. URL `cite-seer.ist.psu.edu/article/david94empirical.html`.
- T. Oates, M. D. Schmill, and P. R. Cohen. Toward natural language interfaces for robotic agents. In *Proc. Agents*, pages 227–228, 2000.
- Jamie Pearson, Jiang Hu, Holly P. Branigan, Martin J. Pickering, and Clifford I. Nass. Adaptive language behavior in hci: how expectations and beliefs about a system affect users’ word choice. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1177–1180, 2006. ISBN 1-59593-372-7.
- C. Raymond Perrault and James F. Allen. A plan-based analysis of indirect speech acts. *American Journal of Computational Linguistics*, 6(3-4):167–182, 1980. URL `cite-seer.ist.psu.edu/perrault80planbased.html`.
- C. Raymond Perrault, James F. Allen, and Philip R. Cohen. Speech acts as a basis for understanding dialogue coherence. In *Proceedings of the 1978 workshop on Theoretical issues in natural language processing*, pages 125–132, Morristown, NJ, USA, 1978. Association for Computational Linguistics. doi: <http://dx.doi.org.proxy.libraries.rutgers.edu/10.3115/980262.980282>.
- Massimo Poesio and David R. Traum. Conversational actions and discourse situations. *Computational Intelligence*, 13(3):309–347, 1997.
- Martha Pollack. A model of plan inference that distinguishes between the beliefs of actors and observers. In Alan W. Biermann, editor, *Proceedings of the 24th Meeting of the Association for Computational Linguistics (ACL)*, pages 207–215, Morristown, New Jersey, 1986. Association for Computational Linguistics. URL `citeseer.ist.psu.edu/pollack86model.html`.
- Martha E. Pollack. Plans as complex mental attitudes. In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 77–103. MIT Press, Cambridge, Massachusetts, 1990.
- Matthew Purver. *The Theory and Use of Clarification Requests in Dialogue*. Ph.D. dissertation, Department of Computer Science, King’s College, University of London, London, 2004.
- Ehud Reiter. NLG vs. Templates. In *Proceedings of the Fifth European Workshop on Natural Language Generation*, pages 95–105, Leiden, The Netherlands, May 1995. URL `cite-seer.ist.psu.edu/reiter95nlg.html`.
- Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA, 2000. ISBN 0-521-62036-8.
- Charles Rich, Candace L. Sidner, and Neal Lesh. Collagen: Applying collaborative discourse theory to human-computer interaction. *Artificial Intelligence Magazine*, 22(4):15–25, 2001.
- Verena Rieser and Oliver Lemon. Learning effective multimodal dialogue strategies from Wizard-of-Oz data: Bootstrapping and evaluation. In *Proceedings of ACL-08: HLT*, pages 638–646, Columbus, Ohio, June 2008. Association for Computational Linguistics.
- D. Roy and A. Pentland. Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1):113–146, 2002.

- Nicholas Roy, Joelle Pineau, and Sebastian Thrun. Spoken dialog management for robots. In *The Proceedings of the Association for Computational Linguistics*, 2000.
- Stephen Schiffer. *Meaning*. Oxford University Press, Oxford, 1972.
- William Schuler, Stephen Wu, and Lane Schwartz. A framework for fast incremental interpretation during speech decoding. *Computational Linguistics*, In Press.
- John Searle. *Speech acts: an essay in the philosophy of language*. Cambridge University Press, 1969.
- Teresa Sikorski and James F. Allen. A task-based evaluation of the TRAINS-95 dialogue system. In *ECAI Workshop on Dialogue Processing in Spoken Language Systems*, pages 207–220, 1996.
- Robert Stalnaker. Assertion. In Peter Cole, editor, *Syntax and Semantics 9*. Academic Press, New York, New York, 1978.
- Robert Stalnaker. Pragmatic presuppositions. In Robert Stalnaker, editor, *Context and Content*, pages 47–62. Oxford, New York, New York, 1974.
- Luc Steels and Tony Belpaeme. Coordinating perceptually grounded categories through language. a case study for colour. *Behavioral and Brain Sciences*, 28(4):469–529, 2005. URL http://www.isrl.uiuc.edu/amag/langev/paper/steels_BBS_color.html.
- Matthew Stone. Specifying generation of referring expressions by example. In *AAAI Spring Symposium on Natural Language Generation in Spoken and Written Dialogue*, pages 133–140, 2003.
- Matthew Stone. Lexicalized grammar 101. In *ACL Workshop on Tools and Methodologies for Teaching Natural Language Processing*, 2002.
- Matthew Stone. Communicative intentions and conversational processes in human-human and human-computer dialogue. In Trueswell and Tanenhaus, editors, *World-Situated Language Use*. MIT, 2004.
- Matthew Stone, Christine Doran, Bonnie Webber, Tonia Bleam, and Martha Palmer. Microplanning with communicative intentions: the spud system. *Computational Intelligence*, 19(4):314–381, 2003.
- William Swartout, Jonathan Gratch, Randall W. Hill, Eduard Hovy, Stacy Marsella, Jeff Rickel, and David Traum. Toward virtual humans. *AI Mag.*, 27(2):96–108, 2006. ISSN 0738-4602.
- Milind Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.
- Joel Tetreault and Diane Litman. Using reinforcement learning to build a better model of dialogue state. In *Proceedings of the 11th Conference of the European Association for Computational Linguistics (EACL)*, 2006.
- Richmond Thomason. Accommodation, meaning, and implicature: Interdisciplinary foundations for pragmatics. In Philip R. Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 325–363. MIT Press, Cambridge, Massachusetts, 1990.
- Richmond H. Thomason, Matthew Stone, and David DeVault. Enlightened update: A computational architecture for presupposition and other pragmatic phenomena. For the Ohio State Pragmatics Initiative, 2006, available at <http://www.research.rutgers.edu/~ddevault/>, 2006.
- Michael Tomasello and H. Rakoczy. What makes human cognition unique? from individual to shared to collective intentionality. *Mind & Language*, 18(2):121–147, 2003.
- Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*, 28: 675–735, 2005.

- D. R. Traum and E. A. Hinkelman. Conversation acts in task-oriented spoken dialogue. *Computational Intelligence*, 8(3):575–599, 1992.
- David Traum. Speech acts for dialogue agents. In M. Wooldridge and A. Rao, editors, *Foundations of Rational Agency*, pages 169–201. Kluwer, 1999a.
- David R. Traum. Computational models of grounding in collaborative systems. In Susan E. Brennan, Alain Giboin, and David Traum, editors, *Working Papers of the AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, pages 124–131, Menlo Park, California, 1999b. American Association for Artificial Intelligence, American Association for Artificial Intelligence.
- David R. Traum. *A Computational Theory of Grounding in Natural Language Conversation*. Ph.D. dissertation, Department of Computer Science, University of Rochester, Rochester, New York, 1994.
- Peter Vanderschraaf and Giacomo Sillari. Common knowledge. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. 2007. URL <http://plato.stanford.edu/archives/fall2007/entries/common-knowledge/>.
- Sebastian Varges and Chris Mellish. Instance-based natural language generation. In *NAACL*, pages 1–8, 2001.
- Sebastian Varges and Matthew Purver. Robust language analysis and generation for spoken dialogue systems. In *Proceedings of the ECAI 2006 workshop on Development and Evaluation of Robust Spoken Dialogue Systems for Real Applications*, 2006.
- J. Williams and S. Young. Scaling pomdps for dialog management with composite summary point-based value iteration (cspbvi). In *Proceedings AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, 2006.
- J. Williams and S. Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422, 2007.
- Jason D. Williams. Demonstration of a pomdp voice dialer. In *Proc Demonstration Session, Annual Meeting of the Association for Computational Linguistics (ACL) with Human Language Technology Conference (HLT)*, 2008.
- Yuk Wah Wong and Raymond Mooney. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of NAACL-HLT*, pages 172–179, 2007.
- Stephen Wu, Lane Schwartz, and William Schuler. Exploiting referential context in spoken language interfaces for data-poor domains. In Jeffrey M. Bradshaw, Henry Lieberman, and Steffen Staab, editors, *Intelligent User Interfaces*, pages 285–292. ACM, 2008.
- Chen Yu and Dana H. Ballard. A multimodal learning interface for grounding spoken language in sensory perceptions. *ACM Transactions on Applied Perception*, 1:57–80, 2004.

Curriculum Vita

David DeVault

Education

2008 Ph.D. in Computer Science, Rutgers University, New Brunswick, NJ, USA

2004 M.A. in Philosophy, Rutgers University, New Brunswick, NJ, USA

2000 B.S. in Engineering and Applied Science, California Institute of Technology, Pasadena, CA, USA

Positions

2007-2008 Visiting Research Assistant, USC Institute for Creative Technologies, Marina del Rey, CA, USA

2003-2006 Graduate Assistant, Department of Computer Science, Rutgers University, New Brunswick, NJ, USA

2003 Intern, Mitsubishi Electric Research Laboratories, Cambridge, MA, USA

2000-2003 Graduate Fellow, Department of Philosophy, Rutgers University, New Brunswick, NJ, USA

2000 Teaching Assistant, California Institute of Technology, Pasadena, CA, USA

Publications

1. DeVault, D. & Bond, A.H.: "A Flexible Eyetracker for Psychological Applications," *Fifth IEEE Workshop on Applications of Computer Vision*, p. 201-206, Palm Springs, California, 2000.
2. DeVault, D. & Stone, M.: "Domain Inference in Incremental Interpretation," *Fourth International Workshop on Inference in Computational Semantics (ICoS-4)*, p. 73-87, Nancy, France, 2003.

3. DeVault, D., Rich, C., & Sidner, C.: "Natural Language Generation and Discourse Context: Computing Distractor Sets from the Focus Stack," *The 17th International Florida Artificial Intelligence Research Society Conference (FLAIRS-2004)*, p. 887-892, Miami Beach, Florida, 2004.
4. DeVault, D. & Stone, M.: "Interpreting Vague Utterances in Context," *The 20th International Conference on Computational Linguistics (COLING 2004)*, p. 1247-1253, Geneva, Switzerland, 2004.
5. DeVault, D., Kariaeva, N., Kothari, A., Oved, I. & Stone, M.: "An Information-State Approach to Collaborative Reference," *The 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), Proceedings Companion Volume, Interactive Poster and Demonstration Sessions*, p. 1-4, Ann Arbor, Michigan, 2005.
6. DeVault, D., Oved, I., & Stone, M.: "Societal Grounding is Essential to Meaningful Language Use," *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, p. 747-754, Boston, Massachusetts, 2006.
7. DeVault, D. & Stone, M.: "Scorekeeping in an Uncertain Language Game," *Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue (SemDial-10)*, p. 139-146, Potsdam, Germany, 2006.
8. Thomason, R., Stone, M., & DeVault, D.: "Enlightened update: A computational architecture for presupposition and other pragmatic phenomena," *Workshop on Presupposition Accommodation, Ohio State Pragmatics Initiative*, Columbus, Ohio, 2006.
9. DeVault, D. & Stone, M.: "Managing ambiguities across utterances in dialogue," *The 2007 Workshop on the Semantics and Pragmatics of Dialogue (DECALOG 2007)*, University of Trento, Italy, 2007.
10. Lee, J., DeVault, D., Marsella, S., & Traum, D.: "Thoughts on FML: Behavior Generation in the Virtual Human Communication Architecture," *The First Functional Markup Language Workshop*, Estoril, Portugal, 2008.
11. DeVault, D., Traum, D., & Artstein, R.: "Practical Grammar-Based NLG from Examples," *The Fifth International Natural Language Generation Conference (INLG 2008)*, Ohio, 2008.
12. DeVault, D., Traum, D., & Artstein, R.: "Making Grammar-Based Generation Easier to Deploy in Dialogue Systems", *The 9th SIGdial Workshop on Discourse and Dialogue (SIGdial 2008)*, Ohio, 2008.
13. Gandhe, S., DeVault, D., Roque, A., Martinovski, B., Artstein, R., Leuski, A., Gerten, J. & Traum, D.: "From Domain Specification to Virtual Humans: An integrated approach to authoring tactical questioning characters", *Interspeech*, Brisbane, Australia, 2008.