Contents

Acknowledgments

A researcher is not a happy researcher if she doesn't love her work. I'd like to start these acknowledgements thanking people that awakened in me the love for the topic of this thesis before even meeting them personally. Thanks to Paul Grice for the inspiring concept of *Implicatures*. Thanks to Richmond Thomason, Matthew Stone and David DeVault for their *Enlightened Update* and their *Tacit acts* and their *Societal Grounding*. Thanks to David Lewis for *Accommodation* and to Herbert Clark for *Bridging*. Thanks to Jerry Hobbs for *Interpretation as Abduction*. Thanks to David Beaver and Henk Zeevat for a deeper understanding of *Accommodation*. Their writings motivated me deeply during different stages of the work behind this thesis. This is also true of my supervisor: I first knew Patrick as a writer. But he also moved me deeply as a speaker (back in 2006 in a hot summer in Málaga), and he's been helping me for three years now as my supervisor and my friend. I'm thankful for each of these different roles that Patrick played in my life.

A researcher is happier if she has a system to play with. I am indebted to Alexander Koller, Yannick Parmentier, Eric Kow and to Ron Petrick for enriching discussions and for their work (FrOz, Tulipa, Geni and PKS). Without them the implementation work behind this thesis would have been impossible.

A researcher is not a happy researcher without a community. I found my community in the *dialogue* people. Thanks to all the people that make SEMDIAL and SIGDIAL possible every year. In particular I want to thank Nicholas Asher (thanks for reading carefully my thesis and giving me motivating comments), Johan Bos (thanks for your e-mails and for your ideas), David DeVault (thanks for the most exciting and laid back discussions I've ever had), Jonathan Ginzburg (thanks for your questions and your interest in my work), Pat Healey (thanks for discussions and pointers to papers), Julia Hirschberg (thanks for your enthusiasm at SIGDIAL '09), Joern Kreutel (thanks for discussions at the EACL demo session) and Ivana Kruijff-Korbayova (thanks for your questions at the University of Saarland) and Matthew Stone (thanks for visiting my home team in Nancy). The list goes on an on with names such as Raquel Fernández, Srini Janarthanam, Anton Benz, Christine Howes, Mary Ellen Foster, Harry Bunt, Staffan Larsson, Gregory Mills, Paul Piwek, and Mark Steedman. I've had the great pleasure of talking to all these people and learning from them. Special thanks go to the people that

organized and participated of the *Young Researchers' Roundtable on Spoken Dialog Systems*. Special thanks go to Alexander Koller, Andrew Gargett, Konstantina Garoufi, Matthew Purver, Ron Petrick and David Schlangen for intensive discussions (from early in the morning till late in the evening) at the *Planning and Grounding meeting*. Finally, I want to thank Bart Geurts for his motivating course at Ealing 2008 and for reading my thesis.

A researcher is not a happy researcher without adventures. In the summer of 2008, I was fortunate to receive an internship position at the Institute for Creative Technologies (ICT) of the University of Southern California in Los Angeles. I'd especially like to thank David Traum for his guidance, as well as ICT's dialogue people such as Ron Artstein, Antonio Roque, Jonathan Gratch and Arno Hartholt for discussions. Special thanks go to the people in the Graphics Lab in ICT who made my stay in Los Angeles be lots of fun!

A researcher is not a happy researcher without a home. During the last three years my research home has been the TALARIS Team at LORIA. As in any home, work and life started to blend together there. For making me feel at home while at work, I want to thank all the members of the TALARIS Team. Particular thanks go to Alexandre Denis (thanks for very motivating discussions and for the translation!) and to Alejandra Lorenzo, Laura Pérez, Lina Rojas Barahona and Katya Lebedeva for being those friends that you can always count on. Guillaume Hoffman, Yannick Parmentier, Sébastien Hinderer, Sergio Mera, Daniel Gorín, Corinna Anderson, Santiago Figueira, Mara Manzano, Paul Bedaride, Ingrid Falk, Marilisa Amoia, Claire Gardent and Tarik Osswald, complete the list of people in my research home. Life in Nancy wouldn't have been the same without the friendship of Humberto Abdelnur, Bruno Woltzenlogel Paleo, Erica Schwindt, Cristian Rosa, Erica Diaz, Diego Caminha and Atif Mashkoor.

A researcher is not a happy researcher without a solid foundation. Thanks to the Universidad del Comahue for my undergrad education. My home university offered me a lot and I look forward to giving them something in return in the years to come. I am particularly grateful to Gerardo Parra for being an excellent person and professor, and to Laura Cecchi for encouraging in me the dream of being a researcher. After flying away from the University of Comahue I completed the European Masters in Computational Logic. All the friends and professors that I met there are part of the person and researcher that I am today. Special thanks to Magda Ortiz de la Fuente for being the best rommie ever.

A researcher is not a happy researcher without dreams. In a few months I will move on to a new home back in Argentina. I want to thanks Laura Alonso Alemany, Gabriel Infante López, Paula Estrella, Agustín Gravano and Franco Luque for making me look forward to that new home.

A researcher is not a happy researcher without love. There is a handful of people that occupy a very special place in my heart. Josefina Sukno, Sofía Marenco and Leticia Gómez, you are my childhood friends with whom I feel today as I felt always, myself. Ale Forquera, Celi Uriz, Jae Yañez and Ale Lorenzo, you've been my best friends for more than ten years now (*las quiero mucho!*). Grandpas, I know that you'd be very proud of your grand daughter

today. Mom, thanks for the limitless love that you have given me. Dad, thanks for the dreams you've shared with me. Mari, thanks for being not only my sister but also my best friend. Gonzi, thanks for being so tender and loving. I know that you four will always be there for me. Carlos, you've given me all the things that I need to be happy: you showed me that work can be like eating chocolate and that computers can be fun to play with, you've accompanied me in many adventures but you've also been that safe home where I can come back and recover, you've understood my dreams and now we can even dream together, you've given all the love that I've needed every day and that's a lot :).

Luciana Benotti
Nancy, France
January, 2010

Chapter 1

What we *don't say* when we say things

One summer afternoon, my mom told my sister: "Buy some food for Tiffy." Then my sister took some money from a kitchen drawer, went to the grocery store near my primary school, bought a pack of low-fat salmon-flavored cat food, and carried it back home. And this is exactly how my mom expected her to act.

Why? Because both of them know that, at home, there is always money in a particular kitchen drawer, that the grocery store near my primary school is the cheapest one, and that Tiffy is our pet cat, who is getting a bit fat and likes salmon. Is that all? Not quite. They also know that in order to buy something you need money, that in order to open a drawer you need to pull it, and many other things that are usually taken for granted (even if you have met neither Tiffy nor my mother).

In this small exchange, my mother and my sister exploited the large amount of information they share in order to leave several actions *unsaid* or, as we prefer to say, in order to leave them *tacit*. **Tacit acts** are actions that we don't explicitly mention, but manage to convey anyway, when we say something.

Now, things would have been quite different if the addressee had not been my sister but my father.

If my sister hadn't been around and my mom had asked my dad instead, she would have said: "Take some money from the kitchen drawer, go to the grocery store near Luciana's school, and buy a pack of low-fat salmon-flavored cat food." Otherwise, he would have gone to the most expensive grocery in town, bought turkey-flavored cat food (which Tiffy stubbornly refuses to eat) and paid with his credit card (which costs 10% more).

My mother wanted to get my sister and my father to do exactly the same thing. However, some of the actions that are tacit in the instructions for my sister are explicit in the instructions for my father; we will say that the instructions for my sister have a coarser granularity than the instructions for my father. Two utterances have different **granularity** if both convey the same content, but one makes explicit certain acts that are left tacit in the other. To distinguish them from *tacit* acts, we will call the *explicit* acts **public acts**.

This thesis studies tacit acts and their role in conversation. In Section 1.1, we give intuitions behind tacit acts using the concept of granularity introduced to Artificial Intelligence (AI) by Jerry Hobbs. The goal of this first section is to create fresh intuitions; we intentionally avoid giving references to theoretical frameworks that study the phenomena that we call tacit acts. However, once the intuitions are in place, Section 1.2 fills the bibliographical gaps left by the previous section, linking the intuitions to the notions of conversational implicature, politeness, abduction and speech act theory. Sections 1.3 and 1.4 motivate the two lines of attack used in this thesis to study tacit acts: empirical analysis of a corpus of human-human conversation (Section 1.3), and analysis by synthesis in the setup of a text-adventure game (Section 1.4). We close this chapter, in Section 1.5, with a one-paragraph summary of each chapter and a graph showing their dependences.

1.1 The intuition of granularity

When giving instructions, switching between different granularities (by leaving actions tacit as my mother did in the Tiffy examples) is not merely legitimate, it is pervasive. Moreover, the ability to switch between granularities is not restricted to instructions but seems to be a more general cognitive capacity. As Jerry Hobbs puts it in a key publication:

> *We look at the world under various grain sizes and abstract from it only those things that serve our present interests. Thus, when we are planning a trip, it is sufficient to think of a road as a one-dimensional curve. When we are crossing a road, we must think of it as a surface, and when we are digging up the pavement, it becomes a volume for us. When we are driving down the road we alternate among these granularities, sometimes conscious of our progress along the one-dimensional curve, sometimes making adjustments in our position on the surface, sometimes slowing down for bumps or potholes in the volume. [Hobbs, 1985, p. 1]*

Hobbs's discussion illustrates the granularities that are required by different physical tasks — planning a trip, crossing a road, digging up the pavement, driving down the road. If the task changes, the required granularity changes. In other words, the ability to conceptualize the world at different granularities and to switch between these granularities are fundamental aspects of the intelligent behavior needed to act in the world. Moreover, considerations of granularity become even more important once language is added to the mix. Let's use Hobbs's terminology and apply it to conversation: when talking, we do not make explicit all those things from the world that serve our present interests, but only those that are necessary for the addressees to fill in the details in the way we intend them to. That is, we speak at the granularity required by the conversational situation, and the ability to judge what the required granularity is, is a key component of being a human conversational agent.

Speaking at the required granularity is a complex task, for the required granularity of a conversation is affected both by the granularity that is necessary for the task (in which the conversation is situated) and a coarser granularity at which the speaker may talk, relying on the addressee to fill in whatever is missing. In order to fill in what is missing the addressee will have to reconstruct, from what the speaker said, the lower level of granularity. Hence, switching between different granularities is also essential for interpretation. The ability to switch between granularities is an integral part of the dynamics of any conversation.

In the next two subsections we further develop the intuitions just introduced. In Section 1.1.1 we discuss why and how people break down a complex activity into various grain sizes, or to put it another way we examine how people segment activities. In Section 1.1.2 we start to explore the complex process by which people switch between different granularity levels, arguing that the different grain sizes relate to each other in structured ways.

1.1.1 Segmenting under various grain sizes

The first question we address here is: what might be the psychological reasons for **segmenting** our view of the world at different granularities? Or more simply: why do we segment? Then we speculate on the mechanisms that people use for performing this segmentation. In other words, our second question is: how do we segment?

Why do we segment?

Reacting quickly to changes is good, but anticipating them is even better. For example, when watching a girl wrap a present you can make predictions about what she is going to do next, using previously learned information on how people typically wrap presents. If you follow her actions closely, you can probably hand her some tape when you know she is going to need it, and before she even asks for it. If she starts doing unexpected actions (such as looking at the ceiling) you will probably think that such surprising actions are part of a different activity, unless you later realize that they are actually related and necessary to achieve the goal of having a nicely wrapped present (perhaps it is getting dark and she wants to turn on the ceiling light).

Once you can relate smaller actions to bigger processes you can start to recognize sequences of actions and predict what's going to happen. Segmentation enables us to treat a (potentially complex) sequence of actions as one unit. The better we can segment an activity into explainable and predictable actions, the better we feel we understand it. To put it briefly: we segment because segmentation makes understanding possible.

How do we segment?

It has been observed [Zacks and Tversky, 2001] that when understanding activities, people tend to divide the stream of behavior into smaller units if the activity is unfamiliar. On the other hand, experts in the activity tend to segment it into larger units. Starting from this observation, one plausible explanation of how people segment activities is that they monitor their ongoing

comprehension of the task and segment the activity when their comprehension begins to falter. In other words, people close a segment and start a new one when the predictions they can make about what is going to happen next are no longer accurate. We will thus say that a **segment** is a sequence of explainable and predictable actions. In this *unipersonal* view of segmentation, segmentation and previous knowledge about the task are intrinsically related. The more we know, the more we can explain and the coarser we segment.

The scenario above is suitable when there is just one person trying to make sense of an ongoing activity. However, when two (or more) people interact they also exploit each others abilities to segment, and they are aware that not everybody segments in the same way. As a result, people segment differently in conversation than when they are segmenting solely for themselves. In conversation, even if the speaker can understand a coarser granularity of an activity, he might break the activity into finer units whenever he thinks that the comprehension of the addressee might be stretched. How the speaker will do this depends heavily on the amount of **mutual information** between the speaker and addressee, as made evident by the contrast between the two instructions in our Tiffy example. My mother wanted my sister and my father to go to the grocery store near my primary school; my mother and my sister both know that that's where they can get the best and cheapest food for Tiffy. However my father doesn't know this (he never remembers household details) and thus my mother cannot rely on him to fill this in on his own as intended; she needs to make it explicit. In this *interactional* view of segmentation, segmentation and mutual knowledge about the task are intrinsically related. The more information is mutual with the addressee, the coarser we can talk to him, since we can rely on him to reconstruct the finer level of granularity that we intended.[1]

1.1.2 Switching between grain sizes

In the first Tiffy example, both my mother and my sister had to switch to the appropriate level of granularity for the task at hand. My mom had to switch to the level that was appropriate for giving the instruction to my sister and my sister had to switch to the level that was appropriate for executing the instruction. What would communication look like if people were not able to switch, that is, if they were stuck at one level of granularity? This is the first question that we will address here. Once the need for **switching** is motivated, we will argue that different levels of granularity relate in structured ways. We will illustrate these issues using more Tiffy examples.

What if people were stuck in one level of granularity?

Here we present two examples of how we imagine conversation, if either the speaker or the addressee were not able to naturally and quietly switch between granularities. First, we modify

[1] Estimating the mutual knowledge that a speaker has with an addressee is an extremely complex issue [Clark, 1996]. The picture presented here is oversimplified; I'm doing that on purpose in order to stimulate intuitions about tacitly conveyed material.

our first Tiffy example to illustrate a situation in which the speaker is not leaving actions tacit when she could:

One summer afternoon my mother told my sister: "Open the kitchen drawer, take some money from the drawer, leave the house, walk to the grocery store that is near Luciana's school, buy a pack of low-fat salmon-flavored cat food, pay for the food, and come back home. Don't drop the pack on your way home".

My sister would be puzzled to receive such a request from my mother because she does not expect my mom to use this level of granularity with her. Even if the content conveyed by the different granularities is the same, my sister expects my mother to exploit the information they share. We are so used to people exploiting our capacity to switch granularities that we usually don't realize it; but we certainly notice when they don't.

Second, suppose it's not the speaker but the addressee who is not able to switch between granularities and is stuck at a particular granularity level, namely, the same level as the speaker in the example above. The resulting interaction might go like this:

Mom(1): buy some food for Tiffy
Sister(2): I don't have money
Mom(3): take some from the kitchen drawer
Sister(4): the drawer is closed
Mom(5): open it
Sister(6): ok, it's open
Mom(7): and well? take the money
Sister(8): ok, I have the money
Mom(9): now, buy some food for Tiffy
Sister(10): I'm not at the grocery

Let's read through the dialogue. I imagine that you think that *I don't have money* is a reasonable thing to say in (2). However (because I am so familiar with the "money-drawer" in my parent's house) I find it just as strange as the whole discussion about opening the drawer from (4) to (6). Our different judgments are reasonable though; it's much more probable that you happen to know more about how to deal with closed drawers than where to find pocket money in my parents' house.

Be that as it may, things certainly get strange from (4) on. Reading this, one gets a strong feeling that my sister does not really want to buy the food, that she is not really cooperating. However, this is not how we designed this example; we designed this example to illustrate a case in which the addressee is stuck in one level of granularity and, although she can identify the obstacles for directly executing the instruction, she is not able to find the tacit acts that will overcome them. That is, she cannot reconstruct the intended level of granularity. It is so difficult to imagine that this can happen to a person (but maybe not to a computer?) that our

first reaction is to find another explanation: my sister does not want to buy the food, or she is just messing with my mother.

Summing up, switching granularities is such a basic ability that is hard even to imagine that either the speaker or the addressee can be in the unlikely and unfortunate jam of not being able to switch.

How do the different granularity levels relate?

The question we address here is whether the different levels of granularity relate in structured ways. This question is important because if we manage to find some structure, then we have a way to start modeling how people switch between different granularities.

Zacks and Tversky [2001] describe a number of psychological experiments related to this question. One of these experiments consisted of asking a person observing an activity to segment it twice, on different occasions, once to identify coarse-grained actions and once to identify fine-grained actions. In this experiment, Zacks and Tversky observed that the boundaries of all the coarse-grained actions coincided with boundaries of fine-grained actions. That is, each coarse-grained action subsumes a group of fine-grained actions in a hierarchical fashion. His conclusion was that when observing activities, people spontaneously track the hierarchical grouping of actions; that is, they construct a **hierarchy of actions**.

It is indeed natural to think of such levels as being hierarchically organized, with groups of fine-grained actions clustering into larger units. For example, for my sister, buying food for Tiffy includes taking money from the kitchen drawer and going to the grocery near my primary school. In turn, the action of taking money includes sub-actions such as pulling the kitchen drawer open and actually taking the money. The hierarchy emerges clearly if you represent the instructions and descriptions of the Tiffy example graphically, as is done in Figure 1.1. In the figure, each coarser-grained action subsumes a group of finer-grained actions.

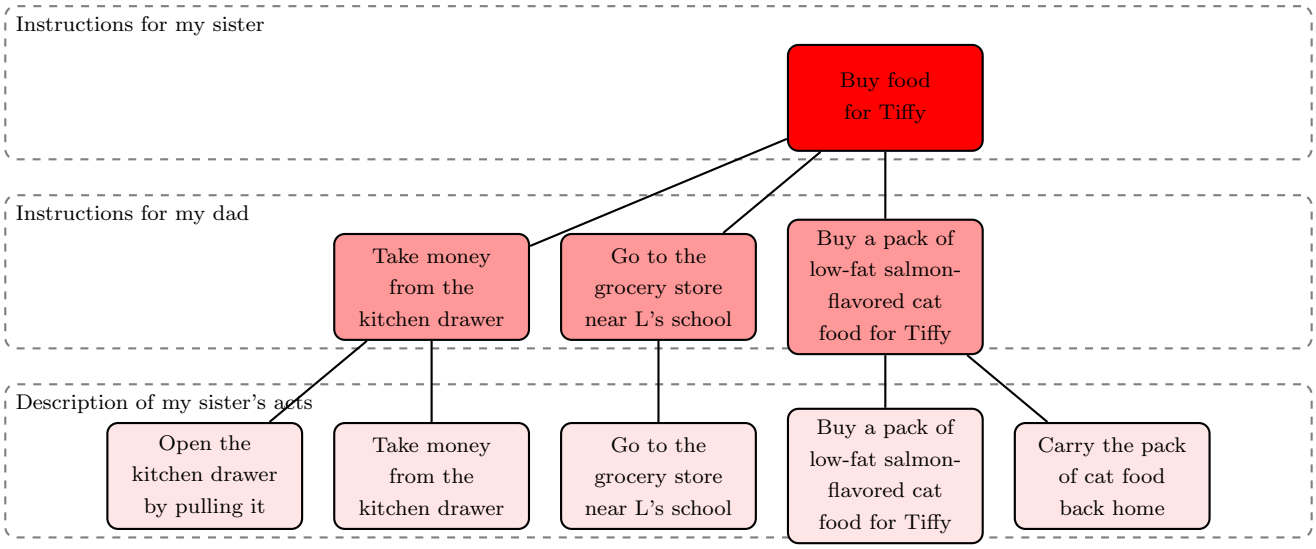


Figure 1.1: Different granularity levels of the Tiffy example

What kinds of relations can we observe between parent and child nodes in this figure? An observation that will turn out to be crucial is that each parent node has a child node whose action is the parent's action. For instance, let's look at the two coarsest levels (the instructions for my sister and the instructions for my dad). The action of the parent node *Buy food for Tiffy* is *Buy* and it has a child whose action is also *Buy*, namely *Buy a pack of low-fat salmon-flavored cat food for Tiffy*. Intuitively, this relation holds between two nodes if the set of words of the child node includes the set of words of the parent node. We will represent this relation by drawing a))))) between a pair of parent-child nodes in the hierarchy and we will call the relation **conceptual strengthening**. Note that conceptual strengthening is transitive. We will call its inverse relation **conceptual weakening**.

Not all the relations that form part of the hierarchy depicted in the figure are of the conceptual strengthening type. For instance, the action of the node *Go to the grocery store near Luciana's school* is not present in the highest level of the hierarchy. Following the terminology we introduced at the beginning of this chapter, we will say that *Take money from the kitchen drawer* and *Go to the grocery store near Luciana's school* are **tacit acts** of the *Buy food for Tiffy* node.

Using the conceptual strengthening relation, we can distinguish between those tacit acts that came before and after the conceptual strengthening relation. We represent those tacit acts that come *before* a conceptual strengthening with the symbol ⌒, and we call them **before tacit acts**. We use ~~~ to represent tacit acts that come *after* a conceptual strengthening, and we call them **after tacit acts**. Summing up, using these three relations (conceptual strengthening, before tacit act, and after tacit act) we can classify all the relations in Figure 1.1 and get Figure 1.2.

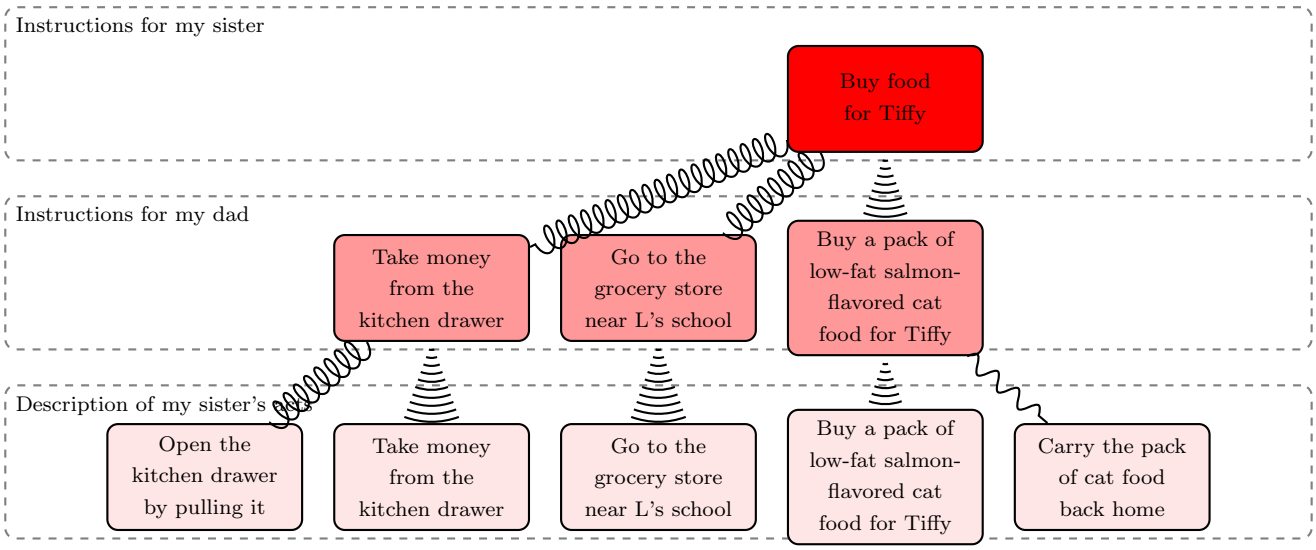


Figure 1.2: Distinguishing the relations in the hierarchy

From these observations we hypothesize that in conversation, speakers switch to higher levels of granularity not by constructing a whole new compact description of the fine-grained

level, but by making explicit a (conceptually weakened) part of the fine-grained level. The part that is made explicit is intended to activate the rest of the details of the fine-grained level in the addressee's mind (that is, it is intended to activate the rest of the segment). In Figure 1.3 we illustrate in dark red the information that is to be promoted (after being conceptually weakened) to the coarse-grained level made explicit in the instructions for my sister. With her instruction, my mother intended my sister to infer the rest of the information depicted in pale red in the Figure 1.3. That is, the nodes depicted in pale red are the tacit acts performed by my mother which she intended my sister to recognize using the large amount of information that they share.

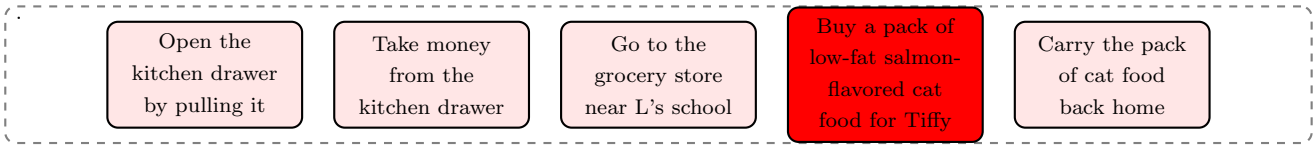


Figure 1.3: Constructing a coarse-grained granularity from a fine-grained one

Summing up, people can productively construct a coarse-level of granularity from a fine-grained one if they are able to view a sequence of fine-grained nodes as constituting a segment. Such a coarse-level can then be used to communicate with other people that know the activity well enough to infer the connection with the (explainable and predictable) tacit acts that form part of the segment.

1.1.3 Where do we go from here?

This account of granularity in conversation leads to two big questions: 1) How do speakers choose the information that they will make explicit in a coarse-grained level; that is, why did my mom choose to say *Buy food for Tiffy* instead of making explicit other parts of the segment? 2) How do addressees reconstruct, from the coarse-grained level that they observe, the fine-grained level that the speaker intends? These two broad questions correspond to the generation of natural language and the interpretation of natural language, respectively, from a granularity perspective. In this thesis we will address the second question; we will concentrate on the problem of how the hearer infers the fine-grained level intended by the speaker.

In order to address this question we will use two methods. On the one hand, we will use an empirical approach in which we look for evidence of the inference of fine-grained-level information that addressees do in human-human dialogue (we motivate this approach in Section 1.3). On the other hand, we will use an approach called analysis by synthesis [Levinson, 1983] in which we build a system prototype that performs the role of the addressee and calculates the fine-grained level; the prototype will be inspired by what was observed in the empirical analysis (we motivate this approach in Section 1.4).

We will introduce in the next section four theoretical frameworks that inform our two methods of study of tacitly conveyed material. In conversational implicature theory (in Section 1.2.1)

we will be looking for surface and functional characteristics of tacitly conveyed material that will help us identify these materials in human-human dialogue (for our empirical analysis study). In Section 1.2.2 we will briefly introduce one of the main pressures that makes conversational partners avoid following a maximally efficient granularity switching strategy in conversation, namely politeness (in the broad sense) as defined by Brown and Levinson [1978]. Studying politeness pressures is not the goal of this thesis, but it is important to be aware of them because they will inevitably appear in our empirical study of human-human data. In Section 1.2.3 we will be looking for characteristics of the reasoning tasks that are traditionally associated with the inference of tacitly conveyed material. Finally, in Section 1.2.4, we will briefly review work which approaches the open problem introduced above from the practical perspective of dialogue systems using concepts from speech act theory; such approaches will motivate and inform the implementation of the prototype done during our analysis by synthesis.

1.2 Looking for knowledgeable advice

In conversation, we happily switch between different granularities and trust that the hearer will be able to fill in the missing details on his own as required. If this is such a common phenomena, then surely we will find a vast literature analyzing it. And so we do, and in a number of different fields. In this section we introduce four theories from this vast literature (drawn from philosophy of language, sociology, logic and computer science) which we find particularly useful for the goals of this thesis.

1.2.1 Philosophical origins of conversational implicature

The notion of **conversational implicature**, which was introduced in a seminal paper by the philosopher of language Grice [1975], is one of the central ideas in philosophy of language. It calls our attention to the fact that many things are meant without being explicitly said, and attempts to explain how this is possible. Let's start with one of the classical examples of conversational implicature from Grice [1975, p. 311].

(1) *Man standing by his car: I am out of petrol.*
Passer by: There is a garage around the corner.

Grice's analysis runs as follows. The utterance made by the passer by (let's call him B) wouldn't have been relevant (to the exchange) if B knew that the garage was closed or that it had run out of petrol. If B is a local person who knows about local garages, it is thus reasonable to assume that B is pointing the man standing by the car (let's call him A) to a garage that is open and currently selling petrol. That is, according to Grice, during the exchange (1), B made the conversational implicature (2):

(2) *The garage is open and has petrol to sell.*

If B had chosen a finer level of granularity for his contribution, the resulting conversation might have been as follows.

(3) *A: I am out of petrol.*
B: There is a garage around the corner. The garage is open and has petrol to sell.

In granularity terms, the exchanges (1) and (3) convey the same information but are realized at different granularity levels. In conversational implicature terms, the exchange (3) *reinforces* the conversational implicature made in (1). Grice's work attempts to characterize the gap between these two granularities utilizing the concept of conversational implicatures. The program underlying this thesis is to use the characteristics of conversational implicatures in order to spot granularity switches in human-human dialogue. So, before going any further, let us explore these characteristics and the terminology used to describe them.

Surface characteristics of conversational implicatures

Defining conversational implicatures and other components of speaker meaning is not an easy task; these are fuzzy concepts and the terminology in the area is far from settled (for discussion see [Potts, 2007]). In what follows, we try to give a coherent characterization of the central properties of conversational implicatures (given the different and contradicting definitions found in [Grice, 1975; Levinson, 1983; Horn, 2004; Potts, 2007]). We illustrate the discussion with Grice's example (1).

Traditionally, conversational implicatures (henceforth CIs) have been characterized using five properties: 1) deniability 2) reinforceability 3) non-lexicality 4) non-detachability 5) calculability. We now discuss these properties in turn.

First, CIs are **deniable**[2] without contradiction. In Grice's example (1), B can append material that is inconsistent with the CI — *but I don't know whether it's open* — and the resulting exchange will not be contradictory.

Second B can add material to the exchange that explicitly asserts the CI — *and I know it's open* — without repeating himself. That is, B can **reinforce** the CI without a sense of repetition. This is what B is doing in example (3).

Third, CIs are **non-lexical**; they do not trace back to lexical items. The CI in Grice's example (1) is not triggered by any particular word in the exchange but is a result of the semantic content of what is said.

Fourth, since a CI is attached to the semantic content of what is said and not to the lexical items, then a CI **cannot be detached** from the utterance simply by changing the words of the

[2]I use the term *deniable* here instead of the most common alternative *cancelable* because I only want to describe with this property cases in which the CI is explicitly denied by the participants of the conversation. Cancelability also include cases in which the CI does not arise because of some feature of the context, which only makes sense in a *defaultism approach* to CIs. In my view, CIs are not associated to sentences by default; all CIs are context-dependent, so if a CI does not arise because of some feature of the context then it simply does not arise and we do not need to say that it was canceled.

utterance by synonyms. B can replace each word in his utterance with a word with the same meaning — he can say *petrol station* instead of *garage* — and the CI will still go through.

Fifth and last, CIs are traditionally considered to be **calculable**. Calculability means that the addressee should be able to infer the CIs of an utterance. For example, in Grice's example (1), A should be able to infer that B conversationally implicates that the garage is open and has petrol to sell.

Let's now see how we can use these five properties to characterize the reinforced material in (3) — *the garage is open and has petrol to sell*. We will start with the first, second and last properties which, in our view, form a natural group.

Deniability and reinforceability are not really two independent properties but two sides of the same coin. CIs are not explicitly said: they are not actual contributions to an exchange, but only potential contributions that the speaker can either deny or reinforce. Now, if we also include in the picture calculability, then we see that not only the speaker but also the hearer can deny or reinforce the CIs. For instance, A can continue Grice's example (1) — *I went there, it's closed* — denying the CI. A can also continue the exchange reinforcing the CI — *oh, and it must be open because it's already 8pm.*

As a consequence, deniability, reinforceability and calculability can be summarized by saying that CIs are **negotiable**. The hearer can infer the CIs of an utterance but cannot be 100% sure that the speaker meant them (and the speaker knows this), so both the speaker and the hearer can further talk about them without getting into a disagreement. That is, hearer and speaker can negotiate the CIs. Negotiability is the characteristic that distinguishes CIs from entailments. **Entailments** (as defined by [Potts, 2007]) include two other components of speaker meaning, namely conventional implicatures and at-issue entailments. Neither type of entailments can be negotiated without a sense of disagreement.

As with deniability and reinforceability, non-detachability and non-lexicality are not really two independent properties. Indeed, non-lexicality can only be tested by evaluating non-detachability. According to Levinson [1983], detachability serves to distinguish CIs from **presuppositions** (a heavily studied component of the speaker meaning). However, non-detachability is problematic because it is difficult to replace all the words in an utterance by so-called synonyms and not change its semantic content. For example, if we look for the word *garage* in a dictionary of synonyms[3] we find synonyms such as *parking lot* and *waiting room*. If we replace *garage* by either of these synonyms, then B's utterance will not implicate that the parking lot or the waiting room has petrol to sell. So if not from a dictionary, where should we take these synonyms from? Who is to decide if two words convey the same meaning or not? These are questions for which (to the best of our knowledge) there is no answer and indeed, strong arguments can be made that no answers are possible [Quine, 1970]. Therefore, the characterization power of non-detachability and non-lexicality remains provisional. At the end of the day, we are left only with negotiability as a test for CIs, and indeed negotiability is the property we will make use of for the empirical study that we introduce in Section 1.3.

[3]We used the Thesaurus.com on-line dictionary of synonyms: http://thesaurus.reference.com/.

A final remark is in order here. Apart from these five characteristics, there is another common classification of CIs that divides them into two groups: those CIs that are **generalized** and those that are **particularized**. However, as some other researchers also do (e.g., Geurts [in press]) we believe that all CIs are context-dependent and thus that all CIs are particularized. Under this view, those CIs that seem to be always present (what many researchers called generalized implicatures) are in fact dependent on context features that happen to be shared by many contexts.

Functional characteristics of conversational implicatures

In this section the underlying question that we address is: what are the roles that CIs play in conversation? Answering this question will help us characterize the inference tasks that we will have to implement during our analysis by synthesis.

Grice claimed that rational communication is governed by a general principle, which he called the cooperative principle.

> ***Cooperative principle:*** *Make your contribution such as is required, at the stage at which it occurs, by the accepted purpose or direction of the talk exchange in which you are engaged. [Grice, 1975, p. 26]*

Much has been said about the relation between CIs and cooperativity (for relevant discussion see Chapter 2). In our view (no matter how cooperative or uncooperative the conversation is) the cooperative principle highlights the fact that a contribution made in a conversation needs to be made so that the hearer is able to anchor it in two directions:

- ***Bridge 1:*** *The contribution has to be anchored into the previous context (i.e., the stage at which the contribution occurs).*
- ***Bridge 2:*** *The relation between the contribution and the goal of the talk (i.e., the accepted purpose or direction of the exchange) has to be worked out.*

In my view, CIs arise as a byproduct of the construction of these two bridges: Bridge 1 anchors the utterance into the previous context, and Bridge 2 anchors the utterance into the goal of the conversation at hand.

This bidirectional perspective also emerges in the Thomason [1990] discussion of the causes of CIs:

- ***Background CIs*** *: Something is said that would be inappropriate if the current presumptions did not involve the implicated propositions. The actual current presumptions may not involve the implicated propositions and the speaker makes the utterance with the intention that the hearer will accommodate it by amending the background presumptions.*
- ***Assertional CIs*** *: What was said has to be mapped to a suitable contribution to the task in which the conversation is situated. It is not the background presumptions but the input utterance which needs revision.*

Arguably, Thomason's classification of CIs causes can be seen as a refinement of the cooperative principle bridges: background CIs correspond to Bridge 1 and assertional implicatures to Bridge 2.

Now, Thomason's twofold classification is a good starting point for answering our question, but is still not concrete enough for our aims, namely, to implement the inference of CIs. It is not concrete enough because in a given dialogue it is difficult to say whether something is a background CI or an assertional CI. For instance, let's go back to Grice's example, which we reproduce here:

(4) *A: I am out of petrol.*
B: There is a garage around the corner.

B's utterance would be inappropriate if he knew that a) the garage is closed or b) has run out of petrol. But does the previous context need to be updated to include a) and b) before updating it with B's utterance, or is it B's utterance which needs to be revised? Well, we think this is a bit like the chicken and egg problem, and crucially depends on the unit of communication that is assumed as basic (proposition, utterance, etc.). What the basic unit of conversation is far from settled, contrary to what is assumed in a large part of the literature in the area. We will not get into the ongoing discussion here (the interested reader is referred to [Wilson and Sperber, 2004]). However, for our purposes the following aspect of this problem is important.

In order to implement the inference of CIs in our analysis by synthesis, we will necessarily have to draw the line between background CI and assertional CI, or as we called them in the previous section, between tacit acts and conceptual strengthening. One important caveat is in order here. The definition of conceptual strengthening given in the previous section is too simplistic: words don't necessarily need to be included for the relation to hold. We illustrated the problem in a simplified way in order to generate the intuition of how coarse-grained levels of granularity can be productively constructed. In reality, however, the problem is much more complex. For instance we would like to say that *We ran out of food for Tiffy* tacitly conveys *Buy food for Tiffy* when uttered by my mother at home. Such cases make evident that the border between tacit acts and conceptual strengthening is fuzzy. We believe that any approach to fix this border can be criticized as arbitrary, and this is then true for our approach. As far as possible then, we will study the tacitly conveyed material without distinguishing between tacit acts and conceptual strengthening. When a distinction is necessary, we will come back and discuss the problems it gives rise to.

The fuzzy distinction between background and assertional CIs is intrinsic to the relevance theoretic analysis [Wilson and Sperber, 2004]. The relevance theoretic interpretation process applies in the same way to resolving indeterminacies and inappropriateness at both the assertional and background level. In this integral approach to interpretation they distinguish between assertional CIs, which they call explicatures, and background CIs, which they call implicated premises and conclusions. In fact, they divide the interpretation task into three levels, which are introduced below.

- ***Explicatures:*** *Constructing an appropriate hypothesis about the conveyed content via decoding, disambiguation, and reference resolution.*
- ***Implicated premises:*** *Constructing an appropriate hypothesis about the necessary contextual assumptions.*
- ***Implicated conclusions:*** *Constructing an appropriate hypothesis about conclusions that are normally drawn from the contribution but not necessary to preserve coherence and relevance.*

This thesis is not written from a Relevance Theoretic perspective, nonetheless we will follow (at least schematically) the ideas just sketched: we will apply a procedural approach to draw a line between explicatures (i.e. assertional implicatures) and implicated material. Explicatures will be inferred using syntactic decoding (parsing), (lexical and syntactic) disambiguation, reference resolution and speech-act-identification (for a definition of this problem see [Jurafsky, 2004]). The difference between implicated premises and implicated conclusions will also be defined in procedural terms in our framework, as we will see in Chapter 4.[4]

Summing up, in our framework we will acknowledge three main roles that CIs play in conversation, namely explicatures (a.k.a. conceptual strengthening in Section 1.1), implicated premises (a.k.a. before tacit acts in Section 1.1), implicated conclusions (a.k.a. after tacit acts in Section 1.1).

1.2.2 From philosophy to sociology: Politeness theory

Politeness theory is the theory that accounts for the strategies that a speaker uses in order to avoid damaging his own face or threatening the addressee when performing a face-threatening acts ("face" is being used here in the sense of social prestige or self image). It was first formulated by Brown and Levinson [1978].

A **face threatening act** is an act that inherently damages the "face" of the addressee or the speaker by acting in opposition to the wants and desires of the other. There are two aspects to face, positive and negative. **Positive face** refers to one's self-esteem, while **negative face** refers to one's freedom to act. The two aspects of face are the basic wants in any social interaction, and so during any social interaction, the participants try not to threaten each others' faces.

Politeness strategies can be seen as soothing layers that are applied to the face threatening act in order to preserve face. The greater the potential for loss of face, the stronger the politeness strategy that should be used. Politeness theory distinguishes between five different politeness strategies:[5]

[4]Intuitively, the difference between implicated premises and implicated conclusions can be explained in terms of the difference between inferences that are necessary to establish coherence and inferences that are elaborative, that is, usual but not necessary (see [Kehler, 2004, p.245]).

[5]The following examples are adapted from [Levinson, 1983] who was using them to exemplify different forms by which the same speech act can be realized.

1. When no soothing layer is put on the act we say that the speaker is performing the act **bald on record** .
   - *Close the door.*

2. The second layer of politeness is to use a **positive politeness** strategy, to try to raise the addressee self-esteem (that is, to claim common ground and use it, or to be friendly).
   - *Do me a big favor, love, and close the door.*
   - *Now, Johnny, what do big people do when they come in?*
   - *Okay, Johnny, what am I going to say next?*

3. The third layer is to use a **negative politeness** strategy which seeks to minimize the imposition on the hearer (that is, giving the hearer an easy way to opt out).
   - *Would you be willing to close the door?*
   - *Would you mind if I was to ask you to close the door?*

4. The fourth layer is called **off-record** and involves being ambiguous in a way that allows the speaker not to assume responsibility for the act.
   - *It's getting cold.*
   - *There is too much noise outside.*

5. If the potential for loss of face is too big, the speaker may make the decision to **abandon** the face threatening act completely and say nothing.

Many of the examples above which are polite (that is, examples in the second, third and fourth level) have been analyzed as conversationally implicating the bald on record contribution *Close the door*. Using our classification from Section 1.2.3 we can say that these conversational implicatures are explicatures that correspond to the procedure of identifying the speech act that has been made (speech acts are discussed in the following section). Then, the process of speech act identification is the process of finding the bald on record contribution that was conveyed. Traditionally, work aiming to characterize this procedure has been called "Interpretation of indirect speech acts" and has been widely studied [Levinson, 1983]. The main goal of this thesis is not to study politeness strategies but we briefly review the work on interpretation of speech acts in Section 1.2.4. The goal of this brief review is to show that there is work which aims to account for explicatures caused by politeness pressures in a broad way, and which will need to eventually be combined with granularity switching in order to have an integral account of conversational implicatures in conversation.

Politeness strategies and granularity switching are two different (but interacting) aspects of human interaction, and both exercise pressures which define the emerging shape of conversation. In this thesis, our main focus are those conversational implicatures that are not related to politeness strategies but to granularity switching. In Chapter 2, we will look for an empirical

setup that highlights CIs related to granularity switching, that is, implicated premises and implicated conclusions.

1.2.3 From philosophy to inference: Abduction

Although there are stirrings of it in Aristotle's work, the modern notion of **abduction** is due to Charles Peirce [reprinted in 1955]. Abduction is a method of logical inference for which the colloquial name is "guessing". The idea is encapsulated in the Piercean abduction schema as follows:

> *The surprising fact Q is observed; but if P were true, Q would be a matter of course. Hence, there is reason to suspect that P is true. [Peirce, reprinted in 1955, p.151]*

According to Jerry Hobbs [2004], the first appeal to something like abduction in natural language understanding was by Grice [1975] when he introduced the concept of CI. Let me remind you once again of Grice's garage example:

(5) *A: I am out of petrol.*
B: There is a garage around the corner.

B's utterance would be inappropriate if he knew that a) the garage is closed or b) has run out of petrol. Then, according to Grice, B made the CI "the garage is open and has petrol to sell (or at least may be open, etc)". Although Grice does not say so, it is pretty much accepted in the area of CIs, that a CI can be seen as an abductive move for the sake of achieving the best interpretation. B said "there is a garage around the corner" because A can probably get petrol there, that is, it is plausible that the garage is open and has petrol to sell.

The fact that Gricean conversational implicatures are abductive inferences have been used to argue that, since abduction is hard then the Gricean picture of conversational implicatures is implausible from a psychological point of view. Two responses can be made to this.

First Geurts [in press] carefully re-examines the evidence that supposedly counts against the psychological plausibility of Gricean inference. He finds it lacking, and indeed concludes that the evidence is on the side of Grice.

So Gricean inference is psychologically plausible, but then how can it be abduction? Our answer will be to point to the existence of an **abduction hierarchy** (see Chapter 3). The thesis uses a point in the hierarchy that is computationally feasible, thus proposing a mechanism that allows us to asset that Gricean inference is psychologically plausible and is a restricted form of abduction.

1.2.4 From philosophy to dialogue systems: Speech acts

An important insight about conversation, due to Austin [1962], is that any utterance is a kind of **action** being performed by the speaker. Austin calls such actions **speech acts** and argues that, in performing a speech act, the speaker is performing three different acts. For example, by telling you "Don't go into the water" we perform three acts:

- A **locutionary act** with distinct phonetic, syntactic and semantic features.
- A **illocutionary act**: namely warning you not to go into the water.
- A **illocutionary act**: if you heed my warning we have thereby succeeded in persuading you not to go into the water.

Austin's work and further extensions by Searle [1975] attempted to define a theory of language as part of a general theory of action. This theoretical work inspired Allen, Cohen and Perrault, from the dialogue system community, to represent speech acts as planning operators; their work resulted on the influential framework for dialogue management called the **BDI model** (Belief, Desire, Intention model) [Allen and Allen, 1994]. The BDI model is a computational model of the problem of determining, given an utterance, which speech act this the utterance realizes. This problem is complex because many (or even most) sentences do not seem to have the speech act associated with their syntactic form. One example is **indirect requests**, in which what seems on the surface to be a question is actually a polite form of a directive to perform an action (a classical example is *can you pass me the salt?*).

Indirect requests have been widely studied, however there are other even more common cases where the surface form of an utterance doesn't match its speech act form. For example, the most common way to realize a **clarification question** is using declarative form (in [Rieser and Moore, 2005] the authors report that 65% of the clarification questions that they found in an English dialogue corpus are realized in declarative word order). These kind of indirect speech acts motivated a second class of computational models for the interpretation of speech acts: the **cue-based model** [Jurafsky, 2004; Jurafsky and Martin, 2008]. Cue-based models view the surface form of the sentence as a set of cues to the speaker's meaning. For these models, interpreting the speech act of an utterance is a classification task; a task they solve by training statistical classifiers on labeled examples of speech acts.

The BDI model focuses on the kind of rich, sophisticated knowledge that is clearly necessary for building conversational agents that can interact. However, we think that the scope of the reasoning that the BDI model proposes using on this rich knowledge is flawed. In this model, the participants devise plans for each conversation and for each topic inside a conversation, where each plan is designed to achieve a specific goal. Actual conversations pose problems for this view. Although people talk to get things done, they typically don't know in advance what they will actually do. The reason is obvious: they cannot know for sure how other people will react to what they say. Conversations are not planned, they are *opportunistic*. Agents have to know why they are asking questions *at the moment* they ask them, but do not need to plan that they will ask such and such questions before the conversation starts.

The cue-based model focuses on statistical examination of the surface cues to the realization of speech acts. In this model, agents are able to make use of the rich lexical, prosodic, and grammatical cues to interpretation. But the breadth and coverage of this model come at the expense of depth; current algorithms are able to model only very simplistic and local heuristics for cues. Hence, this model is not able to reason about the complex pragmatic and world-

knowledge issues that are clearly necessary for building conversational agents that can interact. However, if the complex pragmatic reasoning that is necessary is divided into assertional CIs and background CIs (as we did in Section 1.2.3) we can view the cue-based model as a promising approach to inferring the assertional CIs while tacit acts take care of the background CIs.

To sum up, the BDI approach takes into account rich pragmatic knowledge but views planning as a one shot process, and thus fails to model the interactive and opportunistic nature of conversation. The cue-based model, on the other hand, is intrinsically more local, and thus compatible with interactive and opportunistic modelling, but fails to take into account the kind of rich pragmatic knowledge that is used by real conversational agents. This thesis combines local opportunistic planning with the use of detailed pragmatic knowledge. It treats CIs as genuinely negotiable, in much the same spirit as Clark and Wilkes-Gibbs [1986] treatment of referring expressions.

1.3 Empirical analysis in situated conversation

By **empirical analysis** we mean the direct exploration of the nature of conversational interaction. Empirical analysis, in my view (and contrary to the view apparently held by many Chomskian linguists), should not only study language use beyond the sentence boundary, but also analyze naturally occurring language use, rather than invented examples. The empirical analysis of conversational implicatures that we have in mind is in the spirit of what today is known as **corpus linguistics** .

There is, however, a problem in applying this empirically attractive methodology: CIs resist corpus studies because they are not explicit — they are simply not there in the corpus. That is probably the reason why, to the best of my knowledge, there are no corpus studies of CIs and the literature is based almost entirely on evidence obtained using the **inference method** (see [Chemla, 2009] as a paradigmatic example). The inference method can be seen as a pragmatic-level analogue of the introspective method, which has been historically used in linguistics and philosophy for obtaining native-speakers' judgments on linguistic examples. However, Geurts and Pouscoulous [2009] have shown that the inference method is a biased tool when it comes to gathering data on CIs. Let's briefly consider Geurts and Pouscoulous's (henceforth G&P) argument.

Experiments in the inference paradigm consists in asking the experiment subject whether he would take (a Dutch equivalent of) the sentence (6a) to imply (a Dutch equivalent of) the sentence (6b).

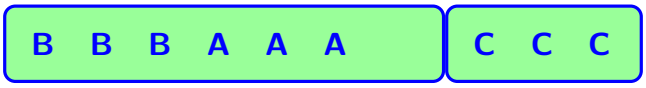
(6) a. *Some of the B's are in the box on the left.*

b. *Not all of the B's are in the box on the left.*

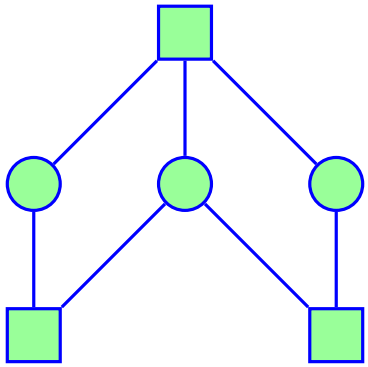
G&P argue that to ask oneself whether or not (6a) implies (6b) is to suggest already that it might be implied. That is, the experiment's question raises the issue of whether or not all of the B's are in the box on the left, and makes it relevant to establish whether this is the case.

G&P claim that such inference experiments do not necessarily tell us anything about how (6a) is interpreted in situations where (6b) is not at stake.

In order to support their claim, G&P compare the inference method with the **verification method** . In the verification version of the previous experiment, subjects have to decide whether (6a) correctly describes the situation depicted below.



Someone who interprets (6a) as implicating (6b) will deny that (6a) gives a correct description of the picture. Participants derived CIs almost twice as frequently in the inference condition (62%) as in the verification condition (34%). In short, the inference task increases the rate of CIs, and the effect is quite substantial. G&P argue that the effect is even more evident in complex sentences such as (7a) describing the situation depicted here:



(7) a. *All the squares are connected with some of the circles.*

b. *All the squares are connected with some but not all of the circles.*

Studies carried out over these kinds of sentences result in participants deriving the CI (7b) from (7a) in 46% of the cases with the inference method, and in 0% of the cases with the verification method. At first sight, this experimental evidence seems to suggest that the verification method is the way to go.

However, in my opinion, if inference methods fail to tell us anything about how utterances should be interpreted when the alleged implicature is *not* at stake, the verification method used by G&P fails in the opposite direction; it fails to tell us anything when the issue *is* at stake. In my view, what G&P's experiments really show is that *whether the issue is at stake or not* is crucial. In other words, they show that even CIs that have been studied as independent from particular characteristics from the context, such as the scalar implicatures carried by utterances like (6a) are, in fact, not independent. Contrary to what many researchers argue (see [Levinson, 2000] for a representative example), scalar implicatures are not generalized, they are particularized and have to be calculated in context.

In conversation, whether a CI is at stake or not is naturally determined by the agreed purpose of the exchange. So our program is to study CIs in their natural context, that is, in natural occurring dialogue.

Studying conversational implicatures in conversation

In Section 1.2.1 we saw that the most clear distinguishing feature of CIs is that CIs are **negotiable**. The hearer cannot be 100% sure that the speaker meant a CI, and the speaker is aware of this, so both of them can further talk about the CI; they can negotiate whether the speaker meant it or not. Conversation provides an intrinsic mechanism for carrying out negotiations of meaning, namely clarifications. In dialogue, Clarification Requests (CRs) provide good evidence of the implicatures that have been made precisely because they make implicatures explicit. Take for instance the CR which can naturally follow Grice's example (1) (first introduced in Section 1.2.1).

(8) *A: we ran out of petrol.*
B: There is a garage around the corner.
A: And you think it's open?

B will have to answer and support the CI — *the garage is open* — if he wants to get it added to the common ground — *Yes, it's open till midnight* — otherwise, if he didn't mean it, he can well reject it without contradiction — *Well, you have a point there, they might have closed.*

My hypothesis is that CIs are a rich source of clarification requests. And my method for verifying the CIs of an utterance will be to infer (some of) the potential CIs of that utterance with respect to a particular context and predict whether the inferred CIs would be clarified or not. Such predictions can then be verified with respect to the actual conversational corpus.

Our program includes looking for CRs that make implicatures explicit in real human-human dialogue. However, the task does not give results as easily as might be thought at first sight. People do not clarify as often as one would expect [Schlangen and Fernández, 2007] and this seems to be related to the fact that clarification interacts with politeness strategies. Accordingly, the corpora in which to perform our empirical analysis will need to take these constraints into account.

The selection of an appropriate corpus, the description of this corpus and the discussion of the CRs that occur there, is discussed in Chapter 2. Chapter 3 discusses different reasoning mechanisms that can be used to infer CIs in conversation using a rich representation of the conversational context. Chapter 4 presents a framework in which we use the reasoning mechanisms discussed in Chapter 3 to infer the CIs made explicit by the CRs in the corpora presented in Chapter 2.

1.4 Analysis by synthesis in a dialogue system

The approach adopted in this thesis for **analysis by synthesis** is the integration of inference methods of CIs into a dialogue system. The inference methods to be integrated are motivated by our empirical study in Chapter 2, our review of available methods in Chapter 3, and our formalization in Chapter 4. The dialogue system that we implement in this thesis should be

able to interpret a natural language utterance, calculate its CIs in a context dependent way, and (taking the CIs into account) decide whether the utterance can be directly added to the conversational context or needs further clarification.

In order to tackle such an ambitious goal, the conversational setup that we will use for our analysis by synthesis is simplified in several ways. To start with, i) the interaction is restricted to speech acts of request which are verbalized as imperatives between two interlocutors. "Request" here is being used in the sense of the first part of the adjacency pair ⟨request, acceptance/clarification/refusal⟩ as defined in [Clark and Schaefer, 1989]. The requests correspond to a set of actions that can be executed in a simulated world; the actions have well defined preconditions and effects. Also, ii) the requests can be issued only by one of the interlocutors (who we will call 'the player'), the other (called 'the game') is limited to accepting (and executing), clarifying or refusing the request. To complete the picture, iii) 'the game' has complete and accurate information about the conversational context (called 'the game scenario'), while 'the player' may have incomplete and even incorrect information.

This conversational setup is formalized in the implementation of a text adventure engine called Frolog which we describe in Chapter 5. Text adventures are computer games that simulate a physical environment which can be manipulated by means of natural language requests (i.e., commands issued to the game). The system provides feedback in the form of natural language descriptions of the game world and of the results of the players' actions. In Frolog, the means-ends inference tasks discussed in Chapter 3 will play the crucial role of coming up with the potential CIs. Frolog includes deterministic and non-deterministic means-ends reasoning capabilities, which we will motivate and describe in Chapter 6. These added inference abilities allow Frolog to discover material left tacit by the player in his requests.

From the late 70s until the earlies 90s, there was a long tradition of using means-ends reasoning mechanisms (in particular, plan recognition) for natural language interpretation in conversation [Allen, 1979; Grosz and Sidner, 1990; Thomason, 1990]. This line of work was eventually abandoned due to its computational complexity. The way in which means-ends reasoning is used in the analysis by synthesis of this thesis is different in subtle and crucial ways from that traditional work. Indeed the system presented in Chapter 6 is intended to be a proof of concept prototype whose goal is to show that means-ends reasoning can be used to infer CIs in a computationally tractable way.

1.5 A map of the thesis

In this chapter, **"What we *don't say* when we say things"**, we presented conversational implicatures intuitively using Jerry Hobbs's broad concept of granularity. Then, we presented conversational implicatures from an interdisciplinary perspective starting from its Gricean origins and moving into: sociology through politeness theory, inference through abduction and dialogue systems through speech act theory. Finally, we motivated the two lines of attack used in this thesis to study conversational implicatures: empirical analysis of a corpus of situated

task-oriented conversation, and analysis by synthesis in the setup of a text-adventure game.

In what follows, we briefly summarize the content of each of the remaining chapters of this thesis and depict their interdependencies in Figure 1.4.

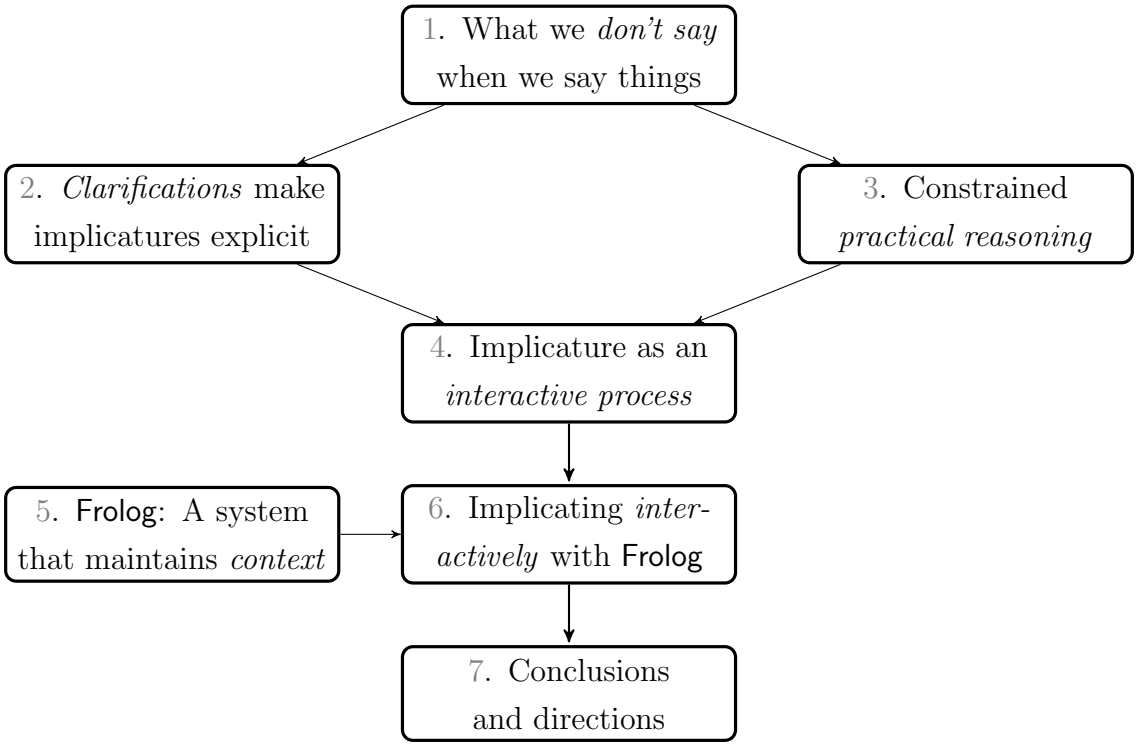


Figure 1.4: Chart of dependencies between chapters

Chapter 2: "*Clarifications* make implicatures explicit" Our hypothesis is that conversational implicatures are made explicit in conversation by clarifications. We start this chapter by reviewing the two lines of analysis of clarifications in the state-of-the-art. The main goal of this review is to construct a definition of the role of clarifications in conversation. In doing so, we identify the features of the corpus that we need for our empirical analysis. Using these features, we select and describe a corpus of dialogues presenting the different kinds of clarifications that we find in the corpus. Finally, we come back to the theoretical concepts introduced in the previous chapter, but now under the light of the empirical evidence found.

Chapter 3: "Constrained *practical reasoning*" In this chapter, we introduce the practical reasoning inference tasks that have been used for interpretation, namely abduction (seminal work by Jerry Hobbs) and plan recognition (seminal work by James Allen). Both tasks are computationally extremely expensive. We then present artificial intelligence planning as a restricted kind of abduction with restricted computational complexity. Planning has been used for natural language generation but not for natural language interpretation; we discuss what kind of inference tasks needed in interpretation are offered by state-of-the-art planners (including non-deterministic ones).

Chapter 4: "Implicature as an *interactive process*" Building upon Chapters 2 and 3, we propose a framework for generating the clarification potential of instructions by inferring its implicatures with respect to a particular context. First, we define the components of the context that are necessary in such a framework. Then, we explain how the restricted kinds of practical reasoning, presented in Chapter 3, can be used in order to infer different kinds of conversational implicatures: implicated premises, implicated conclusions and explicatures. We propose how to use the inferred implicatures in order to predict clarifications. We present the evaluation of this framework analyzing its coverage on the corpus selected in Chapter 2. Moreover, we propose this framework as the basic component of a system which can treat implicature not as one shot process but as an interactive process, generalizing Clark and Wilkes-Gibbs treatment of referring expressions.

Chapter 5: "Frolog: A system that maintains *context*" This chapter presents the general architecture of a dialogue system called Frolog, which implements a text-adventure game. It describes the information resources and the processing modules in detail. The information resources include representations of the common ground, the simulated world in which the dialogue is situated, the act schemes, and an ontology used for deductive inferences in the domain. The processing modules are presented in pairs: parsing/realization, reference resolution/reference generation, other-action execution/own-action determination. The system described cannot infer implicatures and the resulting interactions simulate an addressee that cannot (or will not) accommodate (i.e. bridge) implicatures.

Chapter 6: "Implicating *interactively* with Frolog" In this chapter we describe how Frolog has been extended with bridging abilities. The resulting system is a proof of concept, an analysis by synthesis, that shows that basic bridging abilities can be made in a computationally tractable way. In a first stage, a classical planner can be used in order to implement bridging. Such a system is limited by the complete information assumption made by classical planners. In a second stage, we propose how a planner that reasons on sensing acts (i.e. non deterministic acts) can be used in order to drop the complete knowledge assumption. Then, we present the different kinds of clarification requests the system is able to generate. Moreover, we explain how (a limited kind of) explicatures can be inferred in Frolog setup. We illustrate all the discussion with interactions from the system.

Chapter 7: "Conclusions and directions" We conclude this thesis, summarizing the insights that we believe this thesis gives for the emerging area of computational pragmatics. We discuss the limitations of the model and possible ways to scale it up. We close the thesis with perspectives for the future of the interactive analysis of conversational implicatures. We relate our approach to previous computational work which also implements the treatment of conversational implicatures as an interactive process, we close with ideas for the next step to take.

Chapter 2

Clarifications make implicatures explicit

Conversational implicatures (CIs) are negotiable meanings. Clarification requests (CRs) are the conversational mechanism for negotiating meaning. Therefore, a key hypothesis driving this thesis is that, during a conversation, CRs make CIs explicit.

After having made this observation, the idea of working out what the CIs of an utterance are by exploring its CRs in corpora, is natural — we will refer to this approach as CRs $\rightsquigarrow$ CIs. To do this, it is crucial to delineate the role that the conversational context plays in coming up with the CRs at the point in conversation at which they occur. Moreover, if the link between CIs and CRs is indeed so strong, it should then be straightforward to investigate it from the other direction; namely from CIs to CRs — we will refer to this approach as CIs $\rightsquigarrow$ CRs. My claim is that if all the necessary elements of the context are properly modeled, then the potential CRs can be predicted for each utterance in a dialogue, by inferring its CIs.

This bidirectional method of studying CIs and CRs empirically has a lot to recommend it. It is not only natural but also consistent with the theoretical frameworks reviewed in Chapter 1. But what's more, its predictions can be evaluated by looking at corpora — corpora which contain language occurring in its primary setting: conversation.

In this chapter we explore the method in one direction: CRs $\rightsquigarrow$ CIs. Intuitively, the work done in this chapter is to find out how frequent CRs that make CIs explicit are in conversation. Chapter 4 explores the other direction: CIs $\rightsquigarrow$ CRs. Between these two chapters, Chapter 3 presents the inference tasks that Chapter 4 uses for inferring CIs.

In order to investigate the CRs $\rightsquigarrow$ CIs direction, the program of this chapter is as follows. We delineate the role of CRs in a natural human-human dialogue and characterize those CRs that make CIs explicit in Section 2.1. In Section 2.2, we explore the reasons why CRs are not as frequent in dialogue as one would expect; we discuss here the pressures that make people refrain from requesting clarifications versus those pressures that make them clarify. In Section 2.3, I present a corpus of dialogues in which the characteristics of the interaction force CIs to become explicit in CRs. We explore the different kinds of CRs that we find in this corpus with an extended example. Section 2.4 concludes the chapter, summarizing what we have learned in order to pave the way for the formalized framework (for inferring CIs and predicting CRs) that is presented in Chapter 4.

2.1 The role of clarifications in communication

Practical interest in **clarification requests** (CRs) no longer needs to be awakened in the dialogue research community, as is clear from the amount of recent work on this topic [Gabsdil, 2003; Purver, 2004; Rodríguez and Schlangen, 2004; Rieser and Moore, 2005; Skantze, 2007]. Moreover, in sociolinguistics and discourse analysis, **repair**[1] has been a favored theme for almost three decades now; see [Schegloff, 1987b] as a representative example. However, the theoretical scope of the phenomena and its implications for a theory of meaning are still being delineated. Recently, Jonathan Ginzburg proposed that CRs should be a basic component in an adequate theory of meaning; he summarizes this idea in the following terms:

> *The basic criterion for adequacy of a theory of meaning is the ability to characterize for any utterance type the update that emerges in the aftermath of successful mutual understanding and the full range of possible clarification requests otherwise — this is the early 21st century analogue of truth conditions. [Ginzburg, in press, p.4]*

According to this view, repairs are not a necessary evil but an intrinsic mechanism of language. Repairs, in conversation, are not addressing an error that needs to be solved and forgotten (as most commercial dialogue systems do nowadays) but constitute an intrinsic part of communication. Interpreting an utterance centrally involves characterizing the space of possible requests for clarification of the utterance; that is, interpreting an utterance involves calculating its **clarification potential**. Nonetheless, although we believe that Ginzburg's comment points in the right direction, the idea needs to be made more precise. And giving a precise definition of a CR is more difficult than might be thought at first sight.

Intuitively, we might think that CRs are realized are questions. However, corpus studies indicate that the most frequent realization of CRs is declarative form. Indeed, the form of a CR (although it exhibits some correlations with the CR function [Rodríguez and Schlangen, 2004]) is not a reliable indicator of the role that the CR is playing (as noted below in this section). Neither does it unambiguously indicate whether a dialogue contribution is a CR or not (as the corpus analysis will make evident in Section 2.3).

Two schemes have been proposed in order to characterize CRs according to their function in conversation: one makes central the concept of conversational context while the other focuses on the conversational act. In the rest of this section we introduce these schemes (Sections 2.1.1 and 2.1.2) highlighting their potential contributions for characterizing CIs. Then we discuss a problem left open by these two schemes, namely how to identify CRs in conversation (Section 2.1.3).

[1] For the purposes of this thesis, the terms *clarification* and *repair* can be used interchangeably.

2.1.1 The conversational context in the leading role

The first scheme to classify the function of CRs was proposed by Jonathan Ginzburg, Matthew Purver and colleagues and is presented in a number of publications, the most recent of which is [Ginzburg, in press] (henceforth G&P).

G&P classify CRs functions using the categories shown on Table 2.1.[2] They view CRs as being caused by problems occurring during the anchoring of parameters of utterances into the conversational context.

Category	Problem	Example
Repetition	The hearer cannot identify a surface parameter	What did you say? Did you say 'Bo'?
Clausal confirmation	The hearer finds a problematic value in context (ambiguous or inconsistent) for a contextual parameter	Are you asking if Bo Smith left? You are asking if WHO left?
Intended content	The hearer can find no value for a contextual parameter	What do you mean with pink things? Who is Bo?

Table 2.1: CR classification scheme by Purver and Ginzburg

The G&P classification has been criticized [Rodríguez and Schlangen, 2004; Rieser and Moore, 2005] because, in practice, it seems difficult to decide what the category of a particular CR is; that is, CRs are usually ambiguous in this classification. In fact, G&P recognize this issue and point out that CRs that do not repeat content of the **source utterance** (that is, the utterance that is being clarified) can exhibit all three readings. Notice that these are exactly the CRs that are most interesting for us; the CRs that repeat (part of) the source are probably querying the surface meaning of the source and not its CIs.

However, G&P's classification is only ambiguous if context is not taken into account. It is crucial to analyze the context in order to disambiguate the CR category. Sometimes the immediate linguistic context gives the clue necessary for disambiguation: whereas a repetition reading permits the responder of the CR to repeat her utterance verbatim, a clausal confirmation usually receives a yes/no answer, and an intended content reading requires the responder to reformulate in some way. Hence, the turn of the responder (and the subsequent reaction of the participant originally making the CR) can disambiguate among readings. Consider the following example from [Purver, 2004]. The example shows a case where George's initial clausal interpretation is incorrect (the initiator is not satisfied), and a constituent reading is required (Anon cannot find a contextual value for Spunyarn).

[2] In previous work, what G&P call here *intended content* has been called *constituent reading*, what they call here *clausal confirmation* has been called *clausal reading*, and what they call here *repetition* has been called *lexical reading*. They also define a category that they call *correction reading* but they do not analyze it in detail and we did not find it relevant for this thesis.

George: you always had er er say every foot he had with a piece of spunyarn in the wire
Anon: Spunyarn?
George: Spunyarn, yes
Anon: What's spunyarn?
George: Well thats like er tarred rope

In other situations though, the immediate linguistic context will not be enough (for instance, a reformulation can be a good response to all three types of CRs) and then the whole conversational context might need to be analyzed in order to disambiguate. This makes the G&P's classification difficult to use for annotation studies in which the annotators only get a shallow and localized understanding of the dialogues.

G&P's classification of CRs is relevant for our purposes because it highlights the fact that a mismatch between the contextual requirements of the source utterance (including what the utterance implicates) and the conversational context, is one of the two main causes of CRs (the other has to do with identification of the surface form). If we consider CIs among the contextual parameters of the associated utterance, G&P's analysis gives us a classification of why a CI might be made explicit in a CR. CRs may arise because 1) the inferred CIs are ambiguous or inconsistent with respect to the current conversational context (**clausal confirmation**) or 2) some CI is required in order to anchor the source utterance into the context but it cannot be inferred (**intended content**).[3]

It is worth noticing that G&P's formalization does not include CIs among its contextual parameters. Therefore, the model does not account for CRs at the pragmatic level. Because it doesn't handle CRs at the pragmatic level (and because of the ambiguity problem mentioned before) G&P's analysis has been discarded in recent studies in the dialogue community [Rodríguez and Schlangen, 2004; Rieser and Moore, 2005]. However, if the CIs of an utterance can be inferred in context (remember that CIs are supposed to be calculable), there is no reason why, in principle, they cannot be part of the contextual parameters of an utterance. An initial sketch of how to extend G&P's framework in order to account for CIs is presented in [Ginzburg, in press]. However, Ginzburg argues that there is little empirical evidence for the need for such an extension to their model; he claims that CRs that make explicit CIs are rare in dialogue. We present evidence against this claim in Section 2.2.

2.1.2 The conversational action in the leading role

The second schema of CRs that we present here puts the conversational action in the central role. This schema has been used in recent empirical studies [Gabsdil, 2003; Rodríguez and Schlangen, 2004; Rieser and Moore, 2005]. The classification is based on the four-level model of conversational action independently developed by Jens Allwood [Allwood, 1995] and Herbert Clark [Clark, 1996]. Here, we use Clark's terminology; his model is reproduced in Table 2.2.

[3]The repetition reading is not relevant for us, because in order to infer the CIs of an utterance it is necessary to hear it.

Level	Speaker A's actions	Addressee B's actions
4	A is *proposing* project w to B	B is *up-taking* A's proposal
3	A is *signaling* that p for B	B is *recognizing* that p from A
2	A is *presenting* signal s to B	B is *perceiving* signal s from A
1	A is *executing* behavior t for B	B is *attending* to behavior from A

Table 2.2: Ladder of actions involved in communication

Herbert Clark proposed this model in order to move from Austin's controversial classification of speech acts (see Section 1.2.4) to a **ladder of actions** which characterizes not only the actions that are performed in language use (as Austin's does) but also their inter-relationships.

A ladder of actions is a set of co-temporal actions which has upward causality, upward completion and downward evidence. Let's discuss these three properties in detail using Table 2.2; we will call the speaker Anna and the addressee Barny. Suppose that Anna tells Barny to sit down. Naively viewed, Anna is performing just one action: asking Barny to sit down. It is easy to argue, however, that Anna is in fact doing four distinct, but co-temporal, actions — they begin and end simultaneously. These actions are in a causal relation going up the ladder (from level 1 up to level 4). Anna must get Barny to attend her behavior (level 1) *in order to* get him to hear the words she is presenting (level 2). Anna must succeed at that *in order to* get Barny to recognize what she means (level 3), and she must succeed at that *in order to* get Barny to uptake the project she is proposing (level 4). Summing up, causality (do something *in order to* get some result) goes up the ladder; this property is called **upward causality**.

The ladder can also be used downwards. Observing Barny sitting down (level 4) is good evidence that he recognized what Anna signaled (level 3). But that is also evidence that she got Barny to identify her words (level 2). And it is also evidence that she got him to attend to her voice (level 1). That is, evidence goes down the ladder; this property is called **downward evidence**.

If Barny repeats literally what Anna said (e.g. suppose she spoke in Spanish and then he repeats *sentate*), then Anna has good evidence that he heard what she said (level 2). However, that isn't necessarily evidence that he has recognized what she said; there might be an obstacle in level 3 (for instance, Barny does not know one word of Spanish). If there is such obstacle, she would have completed levels 1 and 2 while failing to complete not only level 3 but also level 4 (it is very unlikely then that Barny sits down after hearing Anna and, even if such a strange coincidence occurs, he will not do it *because* he is up-taking Anna's project). A high level action in the ladder can only be completed by executing all the actions in the lower levels. This property is called **upward completion** .

If you tell somebody something, you expect a reaction from him. If he doesn't answer, you might think he didn't hear you, he doesn't want to answer, or he thinks you are talking to somebody else. None of these situations is very agreeable; we don't like wasting effort. In order not to annoy the speaker, the addressee has two options: either he shows evidence in

level 4 (and then, by downward evidence, the speaker knows that all the levels succeeded), or he indicates the action (in any level) that went wrong. It is not surprising then that languages have intrinsic mechanisms for doing exactly this. CRs are the tools that addressee can use in order to indicate that something went wrong.

The classification, specifying the problems that the addressee can have in the different levels during the interpretation of a conversational action, is summarized in Table 2.3.

Level	Addressee's action	Kind of Problem	Example
4	Uptake	Obstacle for carrying out the proposal	By pressing the red button? And you think it's open?
3	Recognition	Lexical problem Reference problem	What's a "rebreather"? Which button?
2	Perception	Acoustic problem	What did you say?
1	Attention	Establish contact	Are you talking to me?

Table 2.3: CR classification schema by Schlangen and Rodríguez

Let's see where the CRs that make CIs explicit fall in this classification. Intuitively, they will not belong to the lowest levels, that is to levels 1 and 2, because the addressee needs at least to hear the utterance in order to calculate its CIs. In level 3, the classification includes two kinds of problems: lexical and reference problems. One of the characteristics of CIs is that they are not associated with particular words, that is, they are not lexical, so lexical CRs, at least in principle, are not relevant for CIs. The meaning carried by referring expressions has been traditionally associated with presuppositions and not to CIs. It is worth remarking however, that the distinction between CIs and presuppositions is not a settled matter (for discussion see Section 1.2.1). We will come back to this (in Chapter 4) but for the moment we will assume the classical distinction between CIs and presuppositions. Having said this we will restrict ourselves to CRs in level 4 to look for our CIs.

The main problem reported by the researchers using this classification is that, although assigning a level to a CR can be done quite reliably by annotators, they cannot report reliability for the task of identifying CRs in the first place. This difficulty is not superficial and points to the deeper problem of determining the structure of the dialogue. We will discuss the concrete problem of CR identification in the next subsection.

2.1.3 Identification of clarification sub-dialogues

Gabsdil [2003] proposes a test for identifying CRs. Gabsdil's test says that CRs (as opposed to other kinds of contributions in dialogue) cannot be preceded by explicit acknowledgments, as indicated by the following example.

Lara: There's only two people in the class.
a) Matthew: Two people?

b) (??) Matthew: OK. Two people?
(BNC, taken from [Purver et al., 2003])

Gabsdil argues that (a) in the example above *is* a CR because (b) is odd. In (b), Matthew first acknowledges Lara's turn and then shows evidence that the turn contains information that is controversial.[4]

On the other hand, (b) in the example below is fine and then (a) *is not* a CR: the lieutenant acknowledges the sergeant's turn and then he moves to the next topic in the conversation.

Sergeant: There was an accident sir
a) Lieutenant: Who is hurt?
b) Lieutenant: OK. Who is hurt?
(Bosnia scenario, taken from [Traum, 2003])

We think that the test is on the right track. It exploits the fact that positive evidence in one level entails that all the lower levels are complete (downward evidence) and then they may not be clarified. However, the test discards cases that researchers want to treat as CRs, such as the following turn uttered by F (presented by Gabsdil himself as a CR in [Gabsdil, 2003]):

G: I want you to go up the left hand side of it towards the green bay and make it a slightly diagonal line, towards, sloping to the right.
F: Ok. So you want me to go above the carpenter?

More importantly for our purposes, it discards cases in which the CR is making explicit a CI, such as the one in the classical example from Grice (discussed in Chapter 1).

A: I ran out of petrol.
B: There is a garage around the corner.
A: Ok. And you think it's open?

Using the concepts introduced in the previous section it is not difficult to see what's wrong with the test. The problem is that the level of evidence contributed by an acknowledgment is ambiguous. For instance, it can mean *Ok, I heard you* (level 2), *Ok, I identified all the referents and senses of the words of your utterance* (level 3), *Ok, I did it* (level 4). The test needs to be more specific. In particular, in order to entail that all the levels have been successful and then no CR related to any of those is expected, the acknowledgment needs to be replaced by evidence in level 4. And this works for Gabsdil's example.

G: I want you to go up the left hand side of it towards the green bay and make it a slightly diagonal line, towards, sloping to the right.
(??) F: Ok, I did it. So you want me to go above the carpenter?

[4]This could be a felicitous move but requires a very marked intonation or a long pause which would induce some kind of "backtracking" effect.

In this case, *So you want me to go above the carpenter?* is either weird or much more likely to be interpreted as a question about an action that comes *after* having successfully followed G's instruction (that is, as a contribution that is not a CR). Since we do not know this task specification this example seems ambiguous. Which of these two alternatives is actually the case will be determined by the specification of the dialogue task.

Let's look at Grice's example (which probably Grice chose because the underlying task is known to everybody).

A: I ran out of petrol.
B: There is a garage around the corner.
(A goes to the garage and then meets B again)
(??) A: Ok, I got petrol at the garage. And you think it's open?

After giving evidence in level 4 for a contribution, it is really hard to ask a CR about that contribution. The catch is that defining what's evidence in level 4 is not trivial and depends on the context and the conversational act of the source contribution. We will give concrete examples of how to do this in our corpus analysis in Section 2.3. But first we need a corpus; we will explain how we selected one in the next section.

2.2 Looking for the right corpus

We mentioned in Chapter 1 that people do not clarify as often as one would expect [Schlangen and Fernández, 2007]. In particular, Ginzburg [in press] notices that CRs in level 4 are rare in the British National Corpus (BNC corpus) [Burnard, 2000] despite the much higher uncertainty potential than for CRs in level 3 (which are very common in the corpus). In this section, we argue that there are a number of pressures that interact in order to explain the number of CRs that occur in dialogue; we explain why (although it may seem rather counter-intuitive at first sight) it makes sense that too much uncertainty will in fact lower the number of CRs.

We think that the distribution and kinds of CRs found in corpus depend on the characteristics of the task that the dialogues in the corpus are addressing. This might seem a truism, but is a truism widely ignored in the study of CRs; in this area findings in one corpus leads frequently to general claims (e.g., no CRs in level 4 in the BNC corpus lead to the claim that CRs in level 4 are rare in dialogue). Rieser and Moore [2005] recognize a difference between task-oriented and general interest dialogue; there are many different kinds of task-oriented dialogue and we think that a more detailed analysis of the dialogue features and their correlation with the number and kinds of CRs is in order. We investigate such issue in next section in order to select a corpus which is rich in CRs at level 4.

2.2.1 A wish-list of corpus features

G&P did an empirical study [Purver, 2004] on the BNC corpus, which contains English dialogue transcriptions of topics of *general interest*. Based on this experience, G&P report that CRs in

level 4 are rare. This claim motivated two corpus studies in which the authors were looking for CRs in high levels of communication [Rieser and Moore, 2005; Rodríguez and Schlangen, 2004]. Both studies were done on task-oriented dialogue corpora and reported 4th level CRs to be the second or third most common kind of CR (the most common being reference resolution in level 3). We now describe these two studies, highlighting the characteristics of the dialogue task and its relation with the number of CRs

Rieser and Moore [2005] looked for CRs in a corpus of English task-oriented human-human dialogue. The corpus consists of travel reservation dialogues between a client a travel agent. The interactions occur by phone; the participants do not have a shared view of the task. The corpus comprises 31 dialogues of 67 turns each (on average), from which 4.6% of the turns are CRs. 12% of CRs found were classified as level 4 CRs; such as the following:

Client: You know what the conference might be downtown Seattle so I may have to call you back on that.
Agent: Okay. Did you want me to wait for the hotel then?

Rodríguez and Schlangen [2004] looked for CRs in a corpus of German task-oriented human-human dialogue. The dialogues occur in a instruction giving task for building a model plane. The interactions occur face to face; the participants have a shared-view of the task. The corpus consists of 22 dialogues, with 180 turns each (on average), from which 5.8% of the turns are CRs. 22% of CRs found were classified as level 4 CRs; such as the following:

DG: Turn it on.
DF: By pushing the red button?

After evaluating the results from these two studies, we hypothesized that the following characteristics increase the number of CRs in level 4 that can be expected to happen. Hence, we decided that our target corpus should exhibit these characteristics.

Task-oriented: We want dialogues that are task oriented (instead of general interest) because task-oriented dialogues are constrained by the task thus the hearer can have a better hypothesis of what the problem is with the source utterance. He has a motivation for asking for clarifications when the utterance does not fit his model of the task.

Asymmetric knowledge: We want dialogues that are situated in an instruction giving task because then there is asymmetry between the knowledge that the dialogue participants (DPs) have about the task. The Direction Giver (DG) knows how the task has to be done and the Direction Follower (DF) doesn't. Hence, it is to be expected that the DF will have doubts about the task which (both DPs know) can only be answered by the DG. In symmetric dialogues, it might not be clear who has what information and then the DPs might not know who can answer the CRs.

Immediate world validation: We want dialogues that interleave linguistic actions and physical actions because then there is immediate world validation of the interpretations. If an instruction fails in the world, the DF will ask for clarification.

Shared view: We will select a corpus where the DPs have a shared view of the task. In such a setup, the DP that is acting on the world knows that the other participant is observing him and verifying his actions and then will try to be sure of what he has to do before doing it. If he is not sure he will ask.

Long dialogues: We further hypothesized that we need long dialogues (more than 100 turns) because DPs prefer to ask questions when they have a good hypothesis to offer. The longer the interaction, the more background is shared by the DPs and the easier it will be to come up with a good hypothesis.

Punishments: We think that if the DPs are punished for actions that are wrongly performed, then they will clarify more until they are sure of what they have to do.

These properties do not stand on their own but interact in complex ways. We will discuss such interactions as well as how they may inform the theoretical frameworks from Chapter 1 in Section 7.2. Given all these characteristics we selected the SCARE corpus [Stoia *et al.*, 2008] for our empirical study. We describe our findings in this corpus in what follows.

2.2.2 Preliminary findings in the selected corpus

The SCARE corpus consists of fifteen English spontaneous dialogues situated in an *instruction giving task*.[5] The dialogues vary in length, with a minimum of 400 turns and a maximum of 1500; hence, the dialogues are much *longer* than in previous studies. They were collected using the QUAKE environment, a first-person virtual reality game (so there is *immediate world validation*). The task consists of a direction giver (DG) instructing a direction follower (DF) on how to complete several tasks in a simulated game world. The corpus contains the collected audio and video, as well as word-aligned transcriptions.

The DF had no prior knowledge of the world map or tasks and relied on his partner, the DG, to guide him on completing the tasks (so the DPs have *asymmetric knowledge of the task*). The DG had a map of the world and a list of tasks to complete (detailed in Appendix A.2.2). The partners spoke to each other through headset microphones; they could not see each other. As the participants collaborated on the tasks, the DG had instant feedback of the DF's location in the simulated world, because the game engine displayed the DF's first person view of the world on both the DG's and DF's computer monitors (so the DPs *share a view of the task*). Finally, the DPs were *punished* (they were told they would receive less money for performing the experiment) if they pressed the wrong buttons or put things in the wrong cabinets.

We analyzed the 15 transcripts that constitute the SCARE corpus while watching the associated videos to get familiar with the experiment and evaluate its suitability for our purposes.

[5]The corpus is freely available for research in http://slate.cse.ohio-state.edu/quake-corpora/scare/.

Then we randomly selected one dialogue; its transcript contains 449 turns and its video lasts 9 minutes and 12 seconds. Finally, we classified the clarification requests according to the levels of communication using the methodology explained in the Appendix A.

We found 29 clarification requests; so 6.5% of the turns are CRs. From these 29 CRs, 65% belong to the level 4 of Table 2.2, and 31% belonged to level 3 (most of them related to reference resolution). Only 1 of the CRs was acoustic (level 2) since the channel used was very reliable, and another one has to do with establishing contact (level 1).

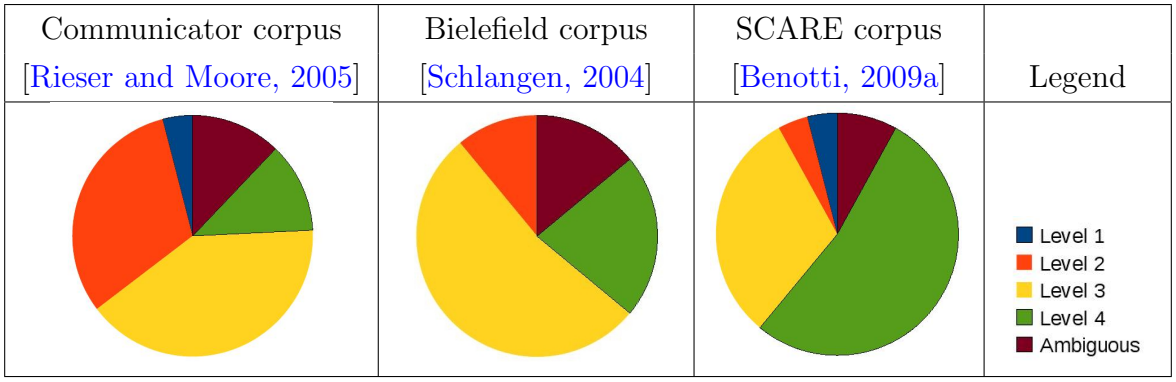


Figure 2.1: Comparing the number of CRs in each level in three corpus studies

In this study, the SCARE corpus seems to present slightly more CRs (a 6.5%) than the corpus analyzed by previous work (which reports that 4%-6% of the dialogue turns are CR). Furthermore, in distinction to the BNC corpus study [Purver, 2004], most CRs in the SCARE corpus occur at level 4. We think that this is a result of the task characteristics mentioned above. Later we will ponder these characteristics, relating them back to the theoretical frameworks introduced in Chapter 1, in Section 7.2. But first we show an extended example of the SCARE corpus and, as promised, we define what's evidence in level 4 for the SCARE task. Using the method for identifying CRs from section 2.1.3 we explore the structure of this SCARE dialogue. We reflect on the relation between the emerging structure of the dialogue and the intuition of granularity from Chapter 1.

2.3 From the corpus back to granularity

In this section we will present an extended sample interaction from the SCARE corpus (which corresponds to one and a half minutes of interaction and 55 turns, in dialogue 3 of the SCARE corpus). The goal of this section is to show the structure of the dialogues that occur in the SCARE corpus. During this dialogue fragment, the dialogue participants were performing the task 3 of the SCARE experiment specified as follows in the instructions that the DG received.[6]

[6]In order to better understand the examples below you may want to read the Appendix A.2 first. The information in the Appendix was available to the participants when they performed the SCARE experiments and it's heavily used in the inferences they draw.

Hide the Rebreather in Cabinet 9. To hide an item you have to find it, pick it up, drop it in the cabinet and close the door. [Stoia, 2007, p. 130]

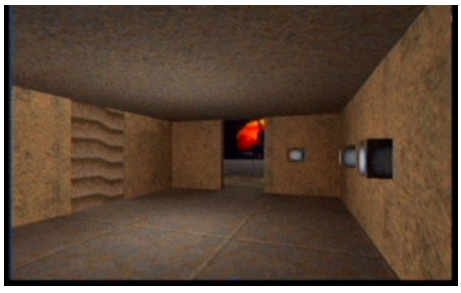
The presentation of this dialogue is divided in the two following subsections. The first gives the warming up necessary for the second. That is, subsection 2.3.1 illustrates the concepts of evidence of proposal and evidence of uptake in level 4 relating them with the identification of CRs. No example of CR in level 4 is presented in this first subsection. Subsection 2.3.2's goal is to illustrate CRs in level 4 as well as their relation with the emergent structure of the dialogue.

2.3.1 Evidence of proposal and evidence of uptake

At the beginning of this dialogue, the DG is instructing the DF to find the rebreather. Let's walk through this part of the dialogue interleaving pictures that show what the DF and the DG were seeing at that moment.[7] The pictures are a screen-shot of the shared view at the moment in which the turns at the right of the picture *start*. The shared view usually changes by the end of the turns because the DF is moving around the world as they talk.

The turns which are evidence of uptake in level 4 are in boldface in the dialogues, and the turns which are evidence of proposal at level 4 are in italics. If evidence for proposal is followed by a turn that is not evidence of uptake (of the proposal) then we say that the turn is a CR and move one tab space inside the conversation structure.

The dialogue fragment reproduced below starts when the DG is trying to get the DF to press the button that is straight ahead in their current view; this button opens the cabinet where the rebreather is located. As part of this project, the DG makes sure first that the DF identifies this button with the sub-dialogue constituted by (1) and (2). Once the button was identified, the short instruction in (3) suffices to convey the goal of this joint project, namely hitting this button; which is up-taken at level 4 in turn (4).



DG(1): see that button straight ahead of you?
DF(2): ***mhm***
DG(3): hit that one
DF(4): ***ok*** **[hits the button]**

Next, the DG moves on to the next project which involves going through the door in the screen-shots below; in order to enter the terrace in which the cabinet that contains the rebreather is located. Here again, the DG first tries to identify the door, however, his turn (5) is a bit misleading because he seems to still be making up his mind about which is the right door. He makes up his mind and utters turn (6) but the DF is already turning left.

[7] It is important to keep in mind that, in the SCARE experiment, the interaction is situated, that is the DG and the DF share a view of the virtual world, hence not only verbal feedback but also visual feedback are important in this setup.

DG(5): now there should be a door
DG(6): straight
DF(7): **[turns left]**

After the DF turns left, he faces the closed door in the screen-shot below and the DG utters (8) to which the DF responds by stopping in (9). In the literature [Gabsdil, 2003], sequences such as those in turns from (5) to (12) are called misunderstandings and corrections and are treated differently than non-understandings and clarifications. However, this distinction does not need to be made when modeling the DF point of view of the interaction. For the DF, corrections that arise from misunderstandings are just further instructions.

DG(8): no wait
DF(9): **[stops]**
DG(10): go around
DF(11): **[turns right]**
DG(12): yeah

After (12), the DG assumes that the goal of identifying the door was achieved (the shared view after (12) is shown in the next figure) and moves to the goal of going through the door with (13).

DG(13): go through there
  DF(14): go through this?
  DG(15): ***yeah***
DF(16): ***ok*** **[goes through door 10]**
DG(17): and then left
DF(18): **[goes left]**

The instruction in turn (13) leads to a CR in (14) which is answered in (15). The instruction (13) is grounded in level 4 in (16) (where the DF executes the action of going through door 10).[8] The next adjacency pair (17) and (18) occurs with no problems.

DG(19): and then pick that up
  DF(20): the rebreather?
  DG(21): ***yeah the rebreather***
DF(22): ***alright*** **[picks up the rebreather]**

[8]The numbers of the doors are taken from the map reproduced in Figure A.2 of the Appendix A

Turn (20) is a CR about the source utterance (19). The CR is resolved and the project proposed by (19) is up-taken in turn (22). With this turn, the DPs achieve the goal of finding and picking up the rebreather.

DG(23): ok go back from where you came
DF(24): [**goes through door 10**]
DG(25): wait
DF(26): [**stops**]

Notice that the structure of the dialogue from turns (1) to (26) is quite flat. From a granularity perspective, the DG is giving the instructions at a low level of granularity and the DF does not have to fill in information on his own. As a result we do not find any CRs in level 4. The CRs that do appear can be classified in level 3 because they are related to reference resolution. We argue that, since the granularity of this dialogue is so low, the DF does not know where the interaction is heading, and thus his predictions about the intended referents are not good. As a result we do observe CRs related to reference resolution. The story is quite different for the dialogue in the next subsection.

2.3.2 Clarifications that make implicatures explicit

In this subsection we have included the annotation of the explicit proposal (between angle brackets) made by each turn (which exhibits evidence of proposal).

The interaction below starts right after the interaction in the previous subsection, that is, the DF has the rebreather and is back inside. Now, the DG utters a instruction in (27). In turn (28) the DF makes explicit a task that must be accomplished *before* putting the rebreather in the cabinet, namely to identify cabinet 9; and by doing so he proposes this task. In turn (29) the DG proposes to identify the cabinet 9 by first identifying its room. Turn (30) is both evidence of uptake of turn (29), that is, the DG answers his own question; but it is also evidence of the proposal ⟨get to the starting room⟩.

DG(27): we have to put it in cabinet nine ⟨put the rebreather in cabinet 9⟩
DF(28): yeah they're not numbered ⟨identify cabinet 9⟩
DG(29): where is cabinet nine? ⟨identify cabinet 9 room⟩
DG(30): ***oh it's kinda like back where you started***
⟨get to the starting room⟩

The DF then hypothesize that in order to get to the starting room he has first to go through the door 11 and asks for confirmation of this hypothesis in turn (31). Turn (32) is confirming this hypothesis and as a result proposing the task ⟨go through door 11⟩ which is up-taken in turn (33).

DF(31): ok so I have to go back through here?
⟨confirm that he has to go through the door 11⟩
DG(32): ***yeah*** ⟨go through door 11⟩
DF(33): **[goes through the door 11]**

Triples of turns (34)-(36), (37)-(39) and (40)-(42) follow the same pattern than the triple (31)-(33). In these turns the DF is reconstructing a low level of granularity (necessary to execute the task) and confirming the necessary steps with the DG. In turn (43) the DF finally gives evidence of uptake of ⟨get to the starting room⟩ proposed in turn (30).

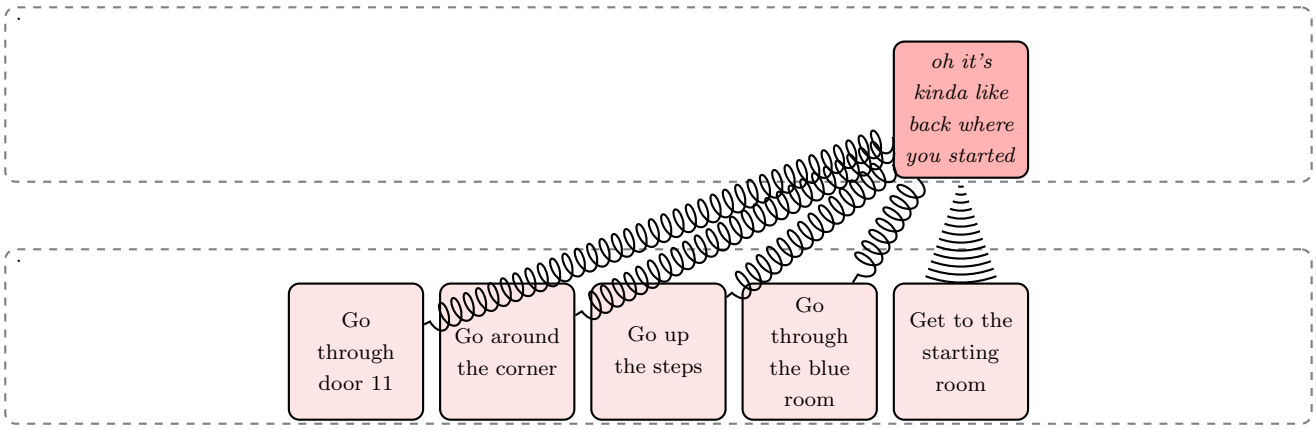
DF(34): and then around the corner?
⟨confirm that he has to go around the corner⟩
DG(35): ***right*** ⟨go around the corner⟩
DF(36): **[goes around the corner]**

DF(37): and then do I have to go back up the steps?
⟨confirm that he has to go up the steps⟩
DG(38): ***yeah*** ⟨go back up the steps⟩
DF(39): alright **[goes up the steps]**

DF(40): and then blue room?
⟨confirm that he has to go through the blue room⟩
DG(41): ***yeah the blue room*** ⟨go through the blue room⟩
DF(42): **[goes through the blue room]**

DF(43): **[gets to the starting room]**
alright and this is more or less where we started

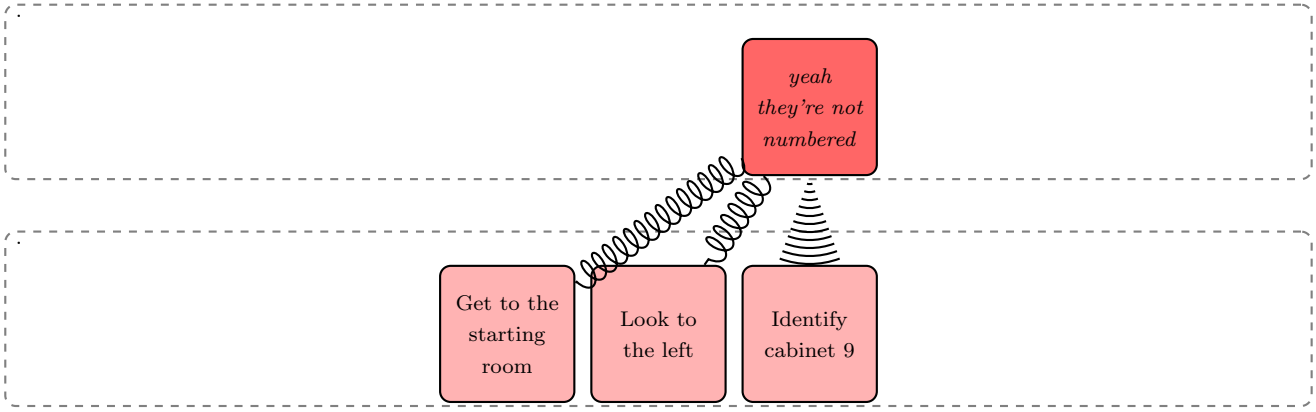
The emergent structure of the dialogue used to achieve the task ⟨get to the starting room⟩, proposed in turn by *"oh it's kinda like back where you started"*, is depicted in the following granularity hierarchy. Using the terminology introduced in Chapter 1 we say that all the nodes connected to *"oh it's kinda like back where you started"* by a ⁀⁀⁀⁀⁀ are its implicated premises. These implicated premises are made explicit in the sub-dialogues (31)- (33), (34)-(36), (37)-(39) and (40)-(42) respectively. The node *"oh it's kinda like back where you started"* is connected by))))) to its explicature ⟨get to the starting room⟩. The project proposed by this explicature is finally closed and grounded in level 4 with *"alright and this is more or less where we started"* in turn (43).



With turn (43) then the task ⟨get to the starting room⟩ is popped from the stack of pending tasks, so the current active task is ⟨*identify cabinet 9*⟩. The DG proposes right away in turn (44) to address this task by first looking to the left. This proposal is grounded in level 4 in turn (45). Then the proposal ⟨identify cabinet 9⟩ made in turn (28) is also grounded in turn (46).

DG(44): *ok so your left cabinet the left one* ⟨look to the left⟩
DF(45): [**looks to the left**]
DF(46): [**identifies cabinet nine**] *alright*

The low level granularity reconstruction elicited by proposal ⟨identify cabinet 9⟩ made by *"yeah they're not numbered"* is depicted as a granularity hierarchy below. ⟨get to the starting room⟩ and ⟨look to the left⟩ are implicated premises of *"yeah they're not numbered"* while ⟨identify cabinet 9⟩ is its explicature. Turn (46) grounds the task ⟨identify cabinet 9⟩ and pops it from the stack.



So the current task is now ⟨put the rebreather in cabinet 9⟩. The DF proposes that the cabinet has to be opened first in turn (47). The DF has a hypothesis of how this can be done and makes it explicit in (48). The hypothesis is confirmed in (49) by what the project of pressing the button is proposed. The project of pressing the button is grounded in turn (50), as a result the cabinet opens.

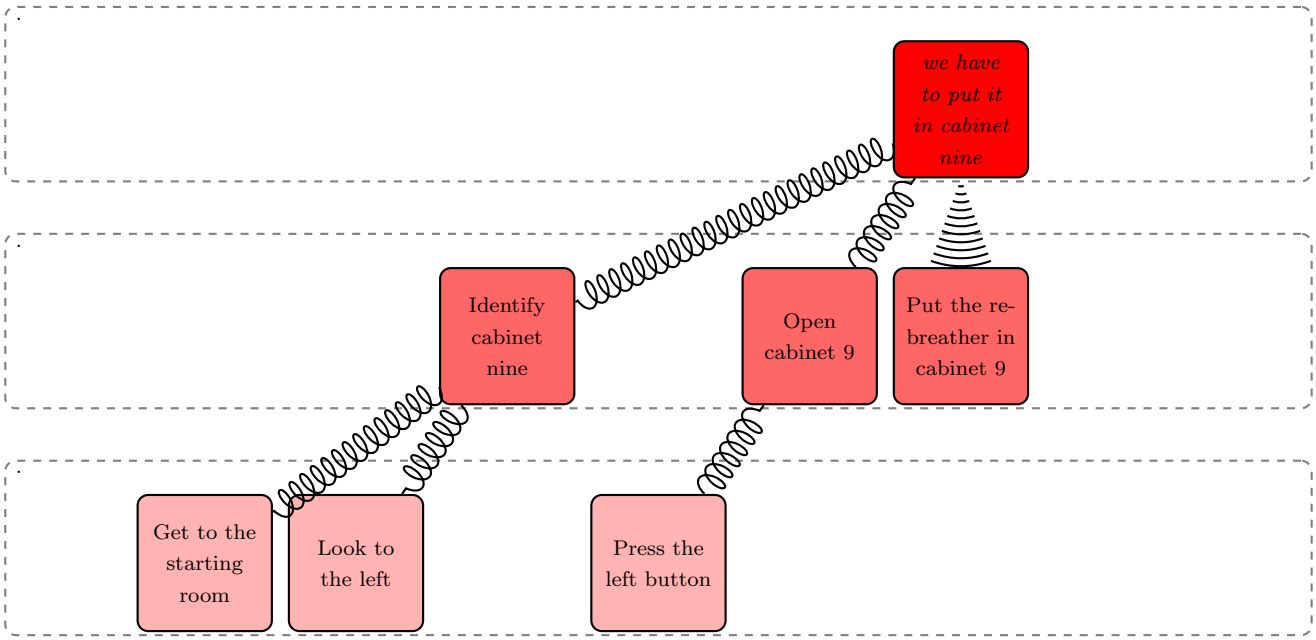


Figure 2.2: The **emergent** structure of *we have to put it in cabinet nine*

DF(47): so how do I open it? ⟨open the cabinet⟩
DF(48): one of the buttons? ⟨confirm that one of the buttons open the cabinet⟩
DG(49): ***yeah*** *it's the left one* ⟨press the left button⟩
DF(50): ***alright makes sense*** **[presses the left button]**
DF(51): **[the cabinet opens]**

Finally the project ⟨put the rebreather in cabinet 9⟩ is grounded in level 4 by the following turn. In this way the initial instruction given by the DG in turn (27) is grounded in level 4. As a result no more clarification requests of (27) can occur.

DF(52): ***alright so we put it in cabinet nine***
[puts the rebreather in cabinet 9]

As this example makes explicit, the implicatures that are finally part of the discourse record are the exclusive responsibility of neither the speaker not the hearer: both are responsible for them and added them throughout the interaction. The **emergent structure** of *"we have to put it in cabinet nine"* is depicted in Figure 2.2.

2.4 Concluding and linking the chapter

In this chapter we explored the idea of working out what the CIs of an utterance are by exploring its CRs in corpora. That is, we explored the CRs $\rightsquigarrow$ CIs direction of the bond between CIs and CRs. We restricted our attention to those CRs that the CR literature classify as 4th level CRs. It is probably fair to say that 4th level CRs are the wastebasket of CRs (just as pragmatics

is the wastebasket of linguistics). That is, once the definition of CR is fixed, then 4th level CRs are roughly those CRs that do not fall in any of the other three levels (see Appendix A for explanation on the methodology used for our annotation of the SCARE corpus). Thus we believe that different kinds of obstacles are put together in level 4; we explore these different kinds of obstacles in Chapter 4.

The restriction that we have imposed on ourselves of looking for CIs only in fourth level CIs may turn out not to be necessary. For instance, we could argue that if the speaker uses a particular referring expression he is implicating the hearer will be able to identify with it the intended referent. That is, the referent is either visible or that the hearer will be able to recover it from memory; and also that the referring expression is able to *univocally* identify the referent. The hearer can negotiate this implicatures with CRs such as *"I cannot see any red button"*, *"Which red button? There are four"*, *"I don't remember any red button"*, etc. In a situated interaction, such as the SCARE experiment, referring acts and other instructions seem more similar than in other discourses. For instance, frequently the DG gives an instruction whose only goal is for the DF to identify some referent, such as *"see that button?"*. Why should this instruction be treated differently than say *"push it"*? That is, why should its implicatures be calculated differently? We have not found one good reason for justifying the difference. However, the main topic of this thesis is not the treatment of referring expressions. There is a large literature in the topic which is beyond the scope of this thesis. Whether the implicatures of referring acts can be treated uniformly with the implicatures of other linguistic acts will remain for further work.

While exploring the CRs $\rightsquigarrow$ CIs direction, we found some stumbling blocks in our way. First, the "rumor" that so called pragmatic CRs (CRs in level 4) are rare in dialogue. We must admit that this rumor is not unmotivated. If CRs at the pragmatic level don't exist then a model of CRs in dialogue can be proposed which does not involve the sort of "deep reasoning" that is computationally very costly, but instead uses some sort of superficial pattern matching. The fact is that in computational approaches to dialogue the word "inference" is not a very popular one as made evident by Purver's words below when talking about their G&P model of CRs.

> *In contrast to plan-based approaches, no inference about plans or intentions behind speech acts is required; and while coercion operations can perhaps be seen as a form of reasoning about context, they are highly constrained and far from general inference.[Purver, 2006, p.26]*

The reason why inference is unpopular in this community is clear. Dialogue system developers need their dialogue systems to interact in real time, they cannot spend precious time (that they also need for tasks such as parsing) in never ending inference processes.

However, as we showed using the SCARE corpus, when the hearer needs to figure out precisely the pragmatic implications of an instruction because they are necessary for the task at hand, he will ask about the implicatures, he will make CRs in level 4. That is to say, the

hearer will clarify when the implicatures are **relevant** to the task at hand. So CRs in level 4 do exist and inference is necessary if we are going to model them. As G&P did with CRs in the lower levels, we believe that constrained and localized — and crucially computationally feasible — inference mechanisms can be used in order to infer the CIs that give rise to CRs in level 4. We explore such mechanisms in Chapter 4 and we integrate them in a "real-time responding" dialogue system in Chapter 6 (Chapter 5 introduces the dialogue system architecture and describe the modules that are not related to CIs).

Another stumbling block was that the definition of CR turned out to be hard to pin down. Our approach to this problem was to use Herbert Clark's action ladder of communication. As a result, given an utterance, the following turn is its CR if it's not evidence of up-taking at level 4. We do not claim here that this is the "right" definition of CR. However, it is good enough for our purposes. We need a definition of CRs that is sufficiently broad so it does not rule out the CRs that we are interested in, that is, CRs at the pragmatic level.

Using this definition of CRs we analyzed in this chapter an extended example of the SCARE corpus. This example illustrate the kind of CIs that are made explicit in CRs in the SCARE corpus. We used the intuition of granularity and the granularity graphs introduced in Chapter 1 to illustrate the CIs found in the SCARE corpus. We believe that this "quasi-systematic' way of identifying CIs in corpora can be of great value to the CI community which desperately needs new and real examples.

In the SCARE corpus, most of the CIs that we found seem to fall in Grice category of **relevance implicatures** . It would be interesting to look for corpora in which other kinds of implicatures, such as **quantity implicatures** are made explicit in CRs.[9] Cross-examination dialogue in a court of law might be good candidates. In such setup, as in the SCARE corpus, the task forces the counselor to make explicit (to put on the record) implicatures that in other (less demanding) setups are taken for granted, as the following exchange set in a cross-examination shows.

C: On many occasions?
W: Not many
C: Some?
W: Yes, a few [Levinson, 1983, p.121]

The "take home message" of this chapter is that the implicatures that are finally part of the discourse record are exclusive responsibility of neither the speaker not the hearer. Both are involved in the process of deciding which CIs will be finally added and they do this through the interaction. Through this interaction the structure of the implicatures that both speaker and hearer are committed to *emerges*.

[9]Preliminary work on conversational implicatures associated with comparative constructions and their interaction with CRs in dialogue was presented in [Benotti and Traum, 2009].

Chapter 3

Constrained *practical reasoning*

Un cronopio pequeñito buscaba la llave de la puerta de calle en la mesa de luz, la mesa de luz en el dormitorio, el dormitorio en la casa, la casa en la calle. Aquí se detenía el cronopio, pues para salir a la calle precisaba la llave de la puerta.

from "Historias de Cronopios y de Famas," Julio Cortázar

Theoretical reasoning tries to assess the way things are. **Practical reasoning** decides how the world should be and what individuals should do. A theoretical proposition is good if it conforms to reality, while a practical proposition has more complex and debatable standards. The distinction between the two can be traced back to the ancient Greek philosopher Aristotle.

The following is an example of practical reasoning due to Aristotle and adapted by Kenny [1966] which illustrates the reasoning process of a physician trying to decide how to heal a man.

This man is to be healed
If (and only if) his humours are balanced, then he will be healed
If he is heated, then his humours will be balanced
If he is rubbed, then he will be heated

So I'll rub him

The physician's reasoning argument could be verbalized as follows: the man can be healed by means of balancing his humours, which can be done by means of heating him, which can be done by rubbing him. Rubbing is in the physician power so he begins his treatment by this, which was the last thing to occur in his practical reasoning. The argument responds to the following pattern: 'H, iff M then H, if T then M, if R then T, so R'. This is the pattern of the inference task known as abduction. This chapter discusses two methods of practical reasoning: abduction and plan-based reasoning; and their application for the inference of conversational implicatures.

Abductive reasoning is usually described as reasoning to the best explanation. This inference mechanism has been extensively used for natural language interpretation in general and for the inference of conversational implicatures in particular. In Section 3.1, we review

the most prominent work in the area, discussing why such a promising approach to natural language interpretation has been largely abandoned; computational complexity is among the most pressing problems.

Plan-based reasoning subsumes two inference tasks: planning and plan-recognition; we present the two tasks formally in Section 3.2 (along with the simplifying assumptions usually made by current reasoners). These two plan-based inference tasks have a long tradition as inference tasks for natural language processing: plan-recognition has been used for natural language interpretation, while planning has been used for natural language generation. Plan-recognition shares with abduction not only its application to natural language interpretation, but also its high computational complexity. Since the early days of artificial intelligence, planning has been used for deciding on future courses of action. Accordingly, in the area of natural language processing, planning has been used almost exclusively in order to decide what to say next (that is, for the task that is known as content planning).

In this thesis, we view planning not as a tool for deciding on future courses of action but as a restricted kind of abductive reasoning: *the observation to be explained is the goal and a possible explanation is the inferred plan.* The idea of using planning instead of abduction is that by restricting the expressivity of the inference task, better computational behavior is obtained. And indeed, planning's computational complexity, though it depends on the language that is used for specifying the planning problem, is computationally cheaper than that of abduction and plan-recognition.

However, planning turns out to be a too restricted inference task, one which makes strong simplifying assumptions such as the assumption of complete information in the states. This assumption means that, if a planner cannot eventually connect all observations with the initial state of the planning problem, it will find no plan. Such an assumption is not made by abduction, which allows information to be added to the states whenever necessary for the sake of finding an explanation. We argue that planning extended with sensing, in order to drop the complete information assumption, is a good comprise between planning and abduction for the task of inferring conversational implicatures.

In the rest of the thesis, we apply planning and planning with sensing to the task of inferring conversational implicatures. Chapter 4 presents the design of the framework and its applications for predicting clarification requests in conversation. And Chapter 6 describe the actual implementation of this framework in a dialogue system that we introduce in Chapter 5. In this chapter we discuss the technical ideas underlying the practical inference reasoning tasks.

3.1 Abductive reasoning

We will begin by introducing the intuitions behind abductive reasoning. Then, I will define the inference task formally. Finally, we will comment on the use of abduction for inferring conversational implicatures.

3.1.1 Intuitions

Abduction, or *inference to the best explanation*, is a form of inference that goes from an observation to a hypothesis that explains the observation; the task was first isolated as an important inference task by Pierce at the end of the 19th century [Peirce, reprinted in 1955]. Medical diagnosis has been a typical illustrative domain for abduction, just as it was for Aristotle's practical reasoning. Obviously, this is a setting that requires specialized knowledge, but abduction also abounds in our day-to-day common sense reasoning. Abduction involves working from evidence to explanation, a type of reasoning characteristic of many different situations with incomplete information encountered in daily life. Pierce describes the process of abduction as follows:

> *The surprising fact Q is observed; but if P were true, Q would be a matter of course. Hence, there is reason to suspect that P is true. [Peirce, reprinted in 1955, p.151]*

As a first approximation then, abduction is a distinctive kind of inference that follows the following pattern, which can be roughly described as Modus Ponens turned backwards:

$$\begin{array}{ll} \text{From:} & Q \\ & \text{if } P \text{ then } Q \\ \hline \text{Infer:} & P \end{array}$$

Let's use a Tiffy example to illustrate this pattern and the kind of inferences that can be made using it.

$$\begin{array}{ll} \text{From:} & \textit{Tiffy is wet} \\ & \text{if } X \textit{ was caught in the rain} \text{ then } X \textit{ is wet} \\ \hline \text{Infer:} & \textit{Tiffy was caught in the rain} \end{array}$$

If Tiffy enters the house wet, then we might assume that she was caught by a sudden rain. However, that's not the only possible explanation. She may have been caught by the neighbour's sprinkler. These competing explanations are obtained using abductive inference. Of course, there may be many possible competing explanations; the conclusions we draw using abduction are only potential explanations and may have to be retracted if we acquire new, contradictory information. Pierce made it clear that it is not reasonable to accept an explanation P obtained by abduction other than as a working hypothesis; such inferences are merely potential and they can be ruled out by new evidence.

3.1.2 The inference task

Definition 1. *We define* ***abduction*** *(that we notate $|\sim$) as the inference task $(T, O)|\sim E$ where:*

- *The* ***theory*** *T is a first order logic ($\mathcal{FOL}$) theory.*
- *The* ***observation*** *O is a set of $\mathcal{FOL}$ ground literals.*

- *The **explanation** E is a set of $\mathcal{FOL}$ ground literals such that:*
  1. $T \cup E \models_{FOL} O$.
  2. $T \cup E$ *is consistent.*

Definition 2. *The **set of explanations** of a theory T and an observation O is* $SE(T, O) = \{E \mid (T, O) \mid\sim E\}$.

Example 3.1. Let's illustrate these definitions with the Tiffy example of the previous section.

- $T_1 = \{\forall X.sprinkled(X) \rightarrow wet(X), \forall X.rained(X) \rightarrow wet(X)\}$
- $O_1 = \{wet(t)\}$
- $SE(T_1, O_1) \supseteq \{\{rained(t)\}, \{sprinkled(t)\}, \{wet(t)\}\}$

The set of explanations contains the explanations that we informally noted in the previous section. It also includes the rather uninformative explanation $wet(t)$; we will talk about that soon.

Now, let's modify the example a little to get a feeling of what abduction is actually inferring. Suppose we looked outside the house and we confirmed that it has not been raining, let's see what our inference task gives us in this case.

Example 3.2. Now, the background theory is different because we know that it has not rained but we still want to explain the same observation, Tiffy is wet. Let's see what explanations we find in this situation:

- $T_2 = T_1 \cup \{\neg rained(t)\}$
- $O_2 = O_1$
- $SE(T_2, O_2) \supseteq \{\{sprinkled(t)\}, \{wet(t)\}\}$
- $SE(T_2, O_2) \not\supseteq \{\{rained(t)\}\}$

Since now we know that it hasn't rained, that explanation is no longer among the possible explanations for Tiffy being wet.

Example 3.2 shows that abduction is actually inferring only potential explanations which may have to be retracted later if we acquire new, contradictory information (e.g. *it has not rained*). That is, with this example, we have shown that abduction is **non-monotonic** .

But what do we do about uninformative explanations such as: *Tiffy is wet because she is wet*. Indeed, although not explicitly stated in the examples above, the sets of explanations also include other uninformative explanations such as *Tiffy is wet because she is wet and it didn't rain*, *Tiffy is wet because she is wet and she wasn't sprinkled*, *Tiffy is wet because she is wet and she was sprinkled and it rained*, etc. This multiplicity of (useless) explanations, which are potentially contradictory among themselves, is a basic problem of abductive inference. Therefore any method for performing abduction must include a method for evaluating and

choosing among alternatives. This can be done either by restricting the literals that can be included in the explanations or by ranking the set of explanations using some criteria. Below we describe the first alternative and in the next subsection we illustrate the second (which has been used when applying abduction to natural language understanding).

The literals that can be included in the explanations can be restricted by an extra input to the inference task as defined below:

DEFINITION 3. *We redefine* ***abduction*** *(that we notate* $\vdash\!\!\sim_a$*) as the inference task* $(T, O, A)\vdash\!\!\sim_a E$ *where:*

- *The* ***abducibles*** *A is a set of* $\mathcal{FOL}$ *ground literals.*
- *The* ***explanation*** *E is a set of* $\mathcal{FOL}$ *ground literals such that it satisfies:*
  1. $(T, O)\vdash\!\!\sim E$
  2. $E \subseteq A$
  3. *E is closed under* $\mathcal{FOL}$ *consequences (in A).*

Using this definition, the set of explanations can be restricted to include only the literals that we are interested in monitoring, by including only those literals in the set of abducibles. For instance, for our running example, if we are only interested in monitoring the literals $sprinkled(t)$ and $rained(t)$ we can restrict the set of abducibles to these literals.

EXAMPLE 3.3. This example applies the Definition 3 to Example 3.2

- $A_3 = \{sprinkled(t), rained(t)\}$
- $(T_2, O_2, A_3)\vdash\!\!\sim_a E$ iff $E = \{sprinkled(t)\}$

With the restricted set of abducibles, the only explanation that is found is that Tiffy has been caught by the neighbour's sprinkler.

3.1.3 Abductive reasoning for implicatures inference

The work applying abduction to natural language understanding (NLU) can be clearly divided into those based on probabilistic methods and those based on symbolic methods. Charniak *et al.* [1989] provide a paradigmatic example of the former, and Hobbs *et al.* [1990] of the later. Both approaches use abduction for extracting as many reasonable and useful inferences as possible from a sentence in a discourse. From the perspective of this thesis, they both propose how to use abduction in order to infer **conversational implicatures (CIs)**.

Charniak *et al.* [1990] propose a method for using **Bayesian networks** for weighting costs in order to associate probabilities with the explanations found. Bayesian networks were first proposed by Pearl [1985] to devise a computational model for human cognition. In [Charniak and Goldman, 1989], the authors apply probabilistic abduction for story comprehension.[1]

[1] A similar approach to interpretation has been recently revived by Henk Zeevat (p.c.) from an optimality theory perspective.

Hobbs *et al.* [1990] propose a way to address the selection of explanations problem on symbolic grounds using a method that they called **weighted abduction**. In weighted abduction the background theory associates costs with their premises. Then the abductive proof process is modified to find the least-cost abductive proof of the goal. The cost of a proof depends not only on the costs associated with the rules; the scoring method favors unifying explanations, that is, premises that explain several different observations at once.[2]

Currently, most researchers in the area of NLU agree that abduction is probably the right theoretical model for inferring CIs. However, there are two major reasons why the approach is not as popular nowadays as was in the 1990s.

One of the objections was raised by Norvig and Wilensky [1990]: these approaches take as their starting point that the hearer must explain why the utterance is true rather that what the speaker was trying to accomplish with it. Utterances are produced to realize a speaker's intention, or more generally, they are actions in the speaker plan to achieve some goal. That is, in order to interpret an utterance, the speaker intention has to be recognized. This objection, even though applicable to both approaches described above, is not intrinsic to using abduction for language interpretation. On the contrary, intention recognition can be easily seen as an abductive inference. When we see another person doing something, we ask ourselves "Given his action (which we have observed) what is he intending to achieve and how?" That is, how can we explain his behavior? So the models discussed here have limitations due to a failure to embrace language as a complex activity, involving actions, goals, beliefs, predictions, and the like, not due to the reasoning task used. Hence, as far as this objection is concerned, abduction is a good idea, it just needs to be used differently.

The second objection is that abductive reasoning is computationally very hard. Existing computational results in the area of abduction are not encouraging. In the full first-order case, abduction is undecidable in general. But although the general case is indeed difficult, it is important to consider the question of how well restricted cases of abduction can be applied in practice before rejecting the method as a viable approach.

Planning promises to solve both of these problems. On the one hand, planning can be viewed as a restricted case of abduction [Eshghi, 1988]: the observation to be explained is the goal and a possible explanation is the inferred plan. This is useful as the computational complexity of planning has been well studied and is much lower than that of abduction [Nau *et al.*, 2004]. On the other hand, the planning inference task was designed, from its origins, for reasoning over actions. An inference task on actions fits perfectly with an intentional view of natural language where utterances are actions in the speaker's plan to achieve some goal.

3.2 Plan-based reasoning

In this section we begin by introducing the intuitions behind plan-based reasoning in Section 3.2.1. Then, in Section 3.2.2, we discuss the typical assumptions about representation

[2]This approach to interpretation is still being actively researched by Jerry Hobbs (p.c.) [Hobbs, to appear].

made by classical plan-based reasoning as well as the representation languages most widely used in the area. Next, we define the two classical inference tasks of plan-based reasoning: planning in Section 3.2.3 and plan-recognition in Section 3.2.4. And we present extensions of classical planning that are relevant for our purposes in Section 3.2.5. Finally, we summarize the long story of the use of these methods in computational pragmatics (in particular, for the inference of conversational implicatures) and I motivate the rather different use we shall make of them in Section 3.3 .

3.2.1 Intuitions

In ordinary English, plans can be many different kinds of things, such as project plans, pension plans, urban plans, and floor plans. In automated-planning research, the word refers specifically to **sequences of action**. **Automated planning** (henforth **planning**) is the inference task of coming up with a sequence of actions that will achieve a goal. Although the use of the word **plan** in the automated planning community has a more restricted sense than the use of the word in English, planners are meant to be used in the real world and there are many everyday problems that can be solved by these planners. We can even see *Tiffy's example* from Chapter 1 as a planning problem. Suppose my mom does not ask my sister to buy food for Tiffy but asks a robot instead. The robot's **goal** will then be to obtain food for Tiffy starting from an **initial state** in which there is no food. The robot should then construct a high level plan which can be described as follows:

Take money from the kitchen drawer, go to the grocery store near Luciana's school, buy a pack of low fat cat food with salmon flavor, and come back home.

We will spare the reader the science fiction details of how the robot will actually be able to perform this tasks in the physical world (the area of robotics is not developed enough to deal with this scenario). But the example might give the reader an intuition of the kind of reasoning that an **automated planner** is able to do. Research in automated planning has always had practical motivations and today is mature enough to be useful in applications that range from game playing to control of space vehicles [Estlin *et al.*, 2003]. In what follows we describe in detail all the components that a planner needs to come up with a plan.

One of the primary parts of the conceptual model of planning is a state-transition system which is a formal model of the real-world system for which we want to create plans.

DEFINITION 4. *A **state-transition system** is a 3-tuple $\Sigma = (S, A, \gamma)$ where:*

- *S is a set of **states** .*
- *A is a set of **actions** .*
- *$\gamma : S \times A \to 2^S$ is a state-transition function. Let $s, s' \in S$ and $a \in A$; if $s' \in \gamma(s, a)$ then the graph contains a **state transition** (that is, an arc) from s to s' that is labeled with the action a.*

A state-transition system my be represented as a directed graph whose nodes are the states in S (for example, see Figure 3.1).

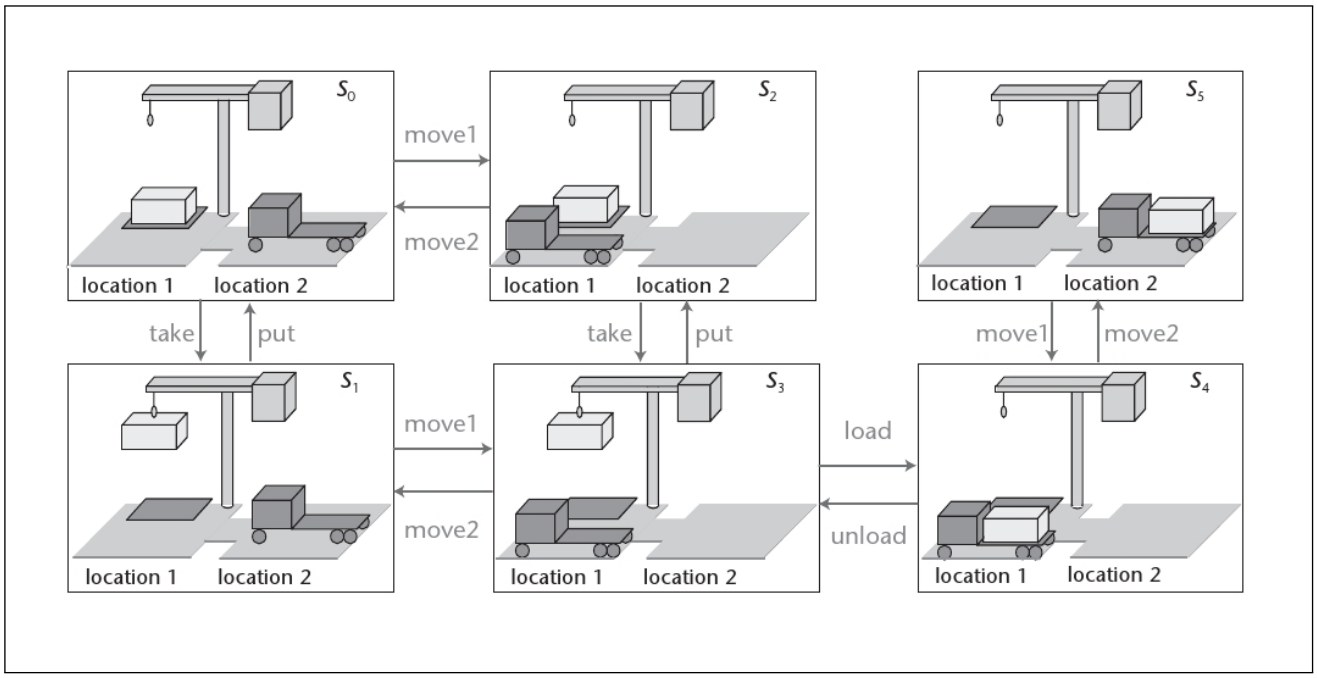


Figure 3.1: A state-transition system for a simple domain

If $a \in A$ and $\gamma(s, a) \neq \emptyset$, then we say that action a is **affordable** in state s. For example, in Figure 3.1, the action load is only affordable in state s_3.

Given a state-transition system Σ, the purpose of planning is to find which actions to apply to which states in order to achieve some objective, when starting from some given situation. The objective can be specified in different ways. The simplest specification consists of a goal state s_g or a set of goal states S_g.[3] For example, if the objective in Figure 3.1 is to have the container loaded onto the robot cart and the cart in location 2, then the set of goal states is $S_g = \{s_5\}$. Now suppose that the initial state is s_0, the task of the planner is to find a plan that takes Σ from s_0 to s_5. An example of such a plan is $\langle \mathsf{take}, \mathsf{move1}, \mathsf{load}, \mathsf{move2} \rangle$. This plan solves this planning problem: that is, if it is executed starting at the initial state s_0, it will take Σ through the sequence of states $\langle s_1, s_2, s_3, s_4, s_5 \rangle$ reaching the expected goal.

Whether to use a sequential plan, or a more general structure, such as a conditional plan, depends on what kind of planning problem we are trying to solve. In the above example, a sequential plan is enough, but more generally, there are some planning problems that cannot be solved by sequential plans. In environments where some of the actions have nondeterministic outcomes, plans are conditional. Such non-sequential plans will turn out to be essential for solving some of the planning problems needed for natural language interpretation.

[3]More generally, the objective might be to get the system into certain states, to keep the system away from certain other states, to optimize some utility function, or to perform some collection of tasks.

3.2.2 Classical plan-based representation

We have provided a high level definition of a planning problem but we haven't yet completely characterized the state transition system yet. The characteristics of this system greatly affect the complexity of the resulting planning problem. In the following subsection, we explore the typical assumptions on Σ made by classical planners and we define the planner's output formally.

Classical assumptions

For almost the entire time that automated planning has existed, it has been dominated by research on domain-independent planning. Because of the immense difficulty of devising a domain-independent planner that would work well in all planning domains, most research has focused on **classical planning** domains, that is, domains that satisfy the following set of restrictive assumptions:

A0) *Finite* Σ: The system Σ has a finite set of states.

A1) *Fully Observable* Σ: Each state $s \in S$ is assumed to have complete information; that is, no state $s \in S$ can represent (either implicitly or explicitly) unknown information.

A2) *Deterministic* Σ: For every state s and action a, $|\gamma(s, a)| \leq 1$. If an action is affordable in a state, its application brings a deterministic system to a unique state.

A3) *Static* Σ: Σ has no external dynamics and there are no unforeseen events. That is, when computing plans, only information in Σ is taken into account (Σ cannot change by external influences).

A4) *Attainment Goals*: The goal is specified as an explicit goal state or a set of goal states S_g. The objective is to find any sequence of state transitions that ends at $s \in S_g$. This assumption excludes, among other things, states to be avoided, constraints on state trajectories, and utility functions.

A5) *Sequential Plans*: A solution plan to a planning problem is a finite sequence of actions which is linearly ordered.

A6) *Implicit Time*: Actions have no duration, they are instantaneous state transitions. This assumption is embedded in the state-transition model, which does not represent time explicitly.

In summary: classical planning requires complete knowledge about deterministic, static, finite systems with restricted goals and implicit time.

Classical languages

Classical planning may appear trivial: planning is simply searching for a path in a graph, which is a well understood problem. Indeed, if we are given the graph Σ explicitly then there is not much more to say about planning. However, it can be shown [Nau *et al.*, 2004] that even in

very simple problems, the number of states in Σ can be many orders of magnitude greater than the number of particles in the universe! Thus it is impossible in any practical sense to list all of Σ's states explicitly. This establishes the need for powerful implicit representations that can describe useful subsets of S in a way that is both compact and can be easily searched.

The set-theoretic representation of classical planning is the simplest. This representation relies on a finite set of proposition symbols that are intended to represent various propositions about the world.

DEFINITION 5. *Let $L = p_1, \ldots, p_n$ be a finite set of propositional symbols. A* ***set-theoretic planning domain*** *on L is a restricted transition-system $\Sigma = (S, A, \gamma)$ such that:*

- *A* ***state*** *s is represented as a collection of propositions; that is, each state $s \in S$ is a subset of L. If $p \in s$, then p holds in the state of the world represented by s, if $p \notin s$ then p does not hold in s.*
- *An* ***action*** *$a \in A$ is represented by giving three subsets of L which we will write as $a = (\text{precond}(a), \text{effects}^+(a), \text{effects}^-(a))$ where:*
  - *The set $\text{precond}(a)$ is called the* ***preconditions*** *of a. An action a is affordable in a state s if $\text{precond}(a) \subseteq s$.*
  - *The set $\text{effects}^+(a)$ is called the* ***positive effects*** *; that is, propositions to assert in s in order to get the resulting state $\gamma(s, a)$.*
  - *The set $\text{effects}^-(a)$ is called the* ***negative effects*** *; that is propositions to retract from s in order to get the resulting state $\gamma(s, a)$.*
- *The* ***state-transition function*** *is $\gamma(s, a) = (s - \text{effects}^-(a)) \cup \text{effects}^+(a)$ if a is affordable in $s \in S$, and $\gamma(s, a)$ is undefined otherwise.*

This representation let us avoid specifying all the states of a planning domain. For example, the state shown in Figure 3.2 might be represented as: { nothing-on-c3, c3-on-c2, c2-on-c1, c1-on-pile1, nothing-on-c8, c8-on-c7, c7-on-c6, c6-on-c5, c5-on-c4, c4-on-pile2, robot-at-loc2, crane-empty }.

In this way, we do not have to specify explicitly all states required by this (very simple) state-transition system; this is fortunate since there are more than two million states (without mentioning their associated transitions). We can just generate them as needed using the transition function specified above.

Even though states can be generated on demand, in the set-theoretic representation, it is still necessary to specify all possible actions explicitly because there is no productive way to generate them. In order to solve this problem, the set theoretic representation has been made more expressive using a notation derived from first-order logic. This representation has also been called STRIPS representation, after an early planning system that used it [Fikes *et al.*, 1972].

DEFINITION 6. *Starting with a function-free first-order language $\mathcal{L}$, a* ***STRIPS planning domain*** *is a transition-system $\Sigma = (S, O, \gamma)$ defined as follows:*

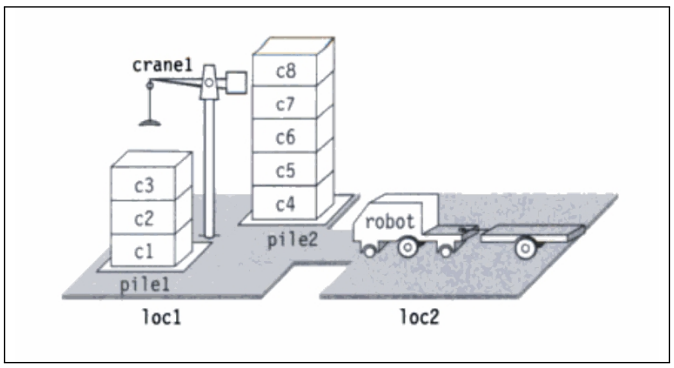


Figure 3.2: A possible state of a planning domain

- *A **state** s is a set of ground atoms in $\mathcal{L}$*
- *An **operator** is a triple $o = (\text{name}(o), \text{precond}(o), \text{effects}^+(o), \text{effects}^-(o))$, $o \in O$, where:*
  - $\text{name}(o)$ *is an expression of the form $n(x_1, \ldots, x_k)$ where n is the operator symbol, and $x_1, \ldots, x_k$ are the arguments, that is, variable symbols that appear anywhere in o.*
  - $\text{precond}(a)$, $\text{effects}^+(a)$ *and* $\text{effects}^-(a)$ *are sets of (ground or non-ground) literals (i.e., atoms and negations of atoms) in $\mathcal{L}$.*
- *The **state-transition function** is defined as in the set-theoretic representation but now instantiating the operators by unifying all its arguments with constant symbols in $\mathcal{L}$ (when an operator is instantiated it is an action as defined in the set-theoretic presentation).*

Example 3.4. Suppose we want to formulate a planning domain in which there are two locations (loc1,loc2), one robot (r1) one crane (crane1), two piles (p1,p2), and three containers (c1,c2,c3). At the bottom of each pile there is a pallet (pallet). Then, the set of constant symbols has ten elements. The state shown in Figure 3.3 is a state in this domain.

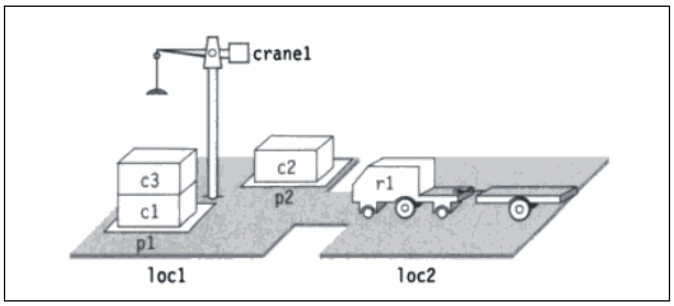


Figure 3.3: The state s_0

This state is represented as follows using the STRIPS representation: $s_0 = \{$attached(p1,loc1), attached(p2,loc1), in(c1,p1), in(c3,p1), top(c3,p1), on(c3,c1), on(c1,pallet), in(c2,p2), top(c2,p2), on(c2,pallet), belong(crane1,loc1), empty(crane1), adjacent(loc1,loc2), adjacent(loc2,loc1), at(r1,loc1), occupied(loc2), unloaded(r1)$\}$

Note that although $\mathcal{L}$ is a first-order logic language, a state is not a set of arbitrary formulas — it is just a set of ground atoms. Both here, and in the set-theoretic representation scheme,

the **closed-world assumption** is made: an atom that is not explicitly specified in a state does not hold in the state.

Let's now specify Σ using the STRIPS representation. To obtain Σ, only the operators need to be specified because the transition function is specified using the operators and the states can be generated using the operators and the transition function (as defined in Definition 6).

PDDL (Planning Domain Definition Language)[4] syntax [Gerevini and Long, 2005] provides formal languages for defining complete planning domains. PDDL defines a set of planning languages with different features and expressive power, such features are specified in the requirements section of the planning problem as exemplified below. The feature strips specifies that the operators are going to be specified as defined in Definition 6.

```
(define (domain dwr)
    (:requirements :strips :typing)
    (:types location pile robot crane container)
    (:objects loc1 loc2 - location r1 - robot crane1 - crane
        p1 p2 - pile c1 c2 c3 pallet - container)
    (:predicates
        (adjacent ?l1 ?l2 - location) (at ?r - robot ?c - location)
        (occupied ?l - location) (belong ?k - crane ?l - location)
        (holding ?k - crane ?c - container) (empty ?k - crane)
        (unloaded ?r - robot) ...)
    (:action move
        :parameters (?r - robot ?from ?to - location)
        :precondition (and (adjacent ?from ?to)
            (at ?r ?from)(not (occupied ?to)))
        :effect (and (at ?r ?to) (not (occupied ?from))
            (not (at ?r ?from))(occupied ?to)))
    (:action load
        :parameters (?k - crane ?c - container ?r - robot l? - location)
        :precondition (and (at ?r ?l) (belong ?k ?l)
            (holding ?k ?c)(unloaded ?r))
        :effect (and (loaded ?r ?c) (not (unloaded ?r))
            (empty ?k)(not (holding ?k ?c))))
    (:action take
        :parameters (?k - crane ?l - location ?c ?b - container ?p - pile) ...)
```

The feature typing indicates that the specification of the planning problem will be **typed**. Typing gives no increase in expressive power of the planning language but enables a planner, which grounds the specification before starting the search, to reduce its search space. For instance, without typing, more than three thousand actions will be instantiated for the operators

[4]PDDL is the standard planning language used for the International Planning Competition since 1998.

above (including obviously impossible actions such as move(loc1,loc1,loc1)); with typing less than one hundred instantiated actions are generated.

Now we are ready to formally define the inference task of planning (Section 3.2.3) and plan recognition (Section 3.2.4) using the STRIPS representation.

3.2.3 The planning inference task

In order to obtain a plan, the planner's input is a planning problem as defined below.

DEFINITION 7. *A **planning problem** is a 3-tuple $P = (\Sigma, s_0, S_g)$ such that:*

- *Σ is a description of the **state-transition system** either using the set-theoretic or the STRIPS representation (that is a set-theoretic or a STRIPS planning domain as defined in Definitions 5 and 6 respectively).*
- *$s_0 \in S$ is an **initial state**.*
- *The set of **goal** states S_g is represented by specifying a set of literals g that have to hold in all states in S_g.*

The output of the planner is a plan. Because of the simplifying assumptions of classical planning, planning reduces to the following problem.

DEFINITION 8. *Given $\Sigma = (S, A, \gamma)$, an initial state s_0, and a subset of goal states S_g, a **plan** is a sequence of actions $\langle a_1, a_2, \ldots, a_k \rangle$ such that the sequence of states $\langle s_0, s_1, \ldots, s_k \rangle$ satisfies $s_1 \in \gamma(s_0, a_1), s_2 \in \gamma(s_1, a_2), \ldots, s_k \in \gamma(s_{k1}, a_k)$, and $s_k \in S_g$.*

EXAMPLE 3.5. Let's take as our Σ the one defined in Example 3.4 and as our initial state the state shown in Figure 3.3. Suppose that the goal of the planning problem is $g_1 =$ {loaded(r1,c3),at(r1,l1)}. This goal holds in the state s_6 depicted in Figure 3.4.

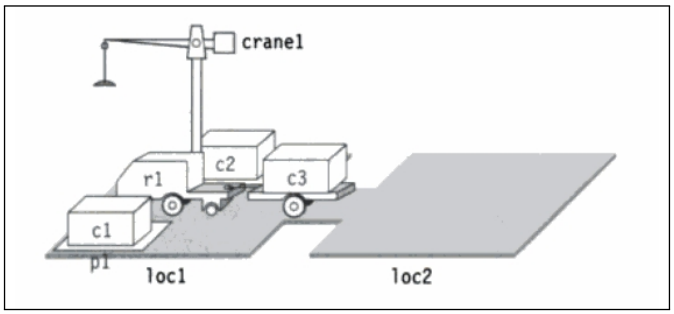


Figure 3.4: The state s_6, the goal state

The plan ⟨take(crane,loc1,c3,c1,p1), move(r1,loc2,loc1),load(crane1,c3,r1,loc1)⟩ solves this planning problem.

The difficulty of planning is dependent on the simplifying assumptions employed which we reviewed in Section 3.2.2, for example, atomic time, deterministic time, complete observability, etc. Classical planners make all these assumptions and have been studied most fully. Some traditional algorithms for implementing the classical inference task of planning include: forward chaining and backward chaining state-space search, and search through plan space. Some famous algorithms that dramatically advanced the area of planning include: translation to propositional satisfiability (for example, SATPLAN [Kautz and Selman, 1992]) the use of relationships among conditions (for example, GRAPHPLAN [Blum and Furst, 1997]), and heuristic search (for example, FAST-FORWARD [Koehler and Hoffmann, 2000]).

There are many extensions to classical planning that drop various classical assumptions; we explore one of them in Section 3.2.5. These extensions are very much current research and they are not as well understood as classical planning. The decision of which planner to use will depend on the problem that we need to model (and also, on the performance requirements of our application).

However, before turning to such extensions of the planning task let us first describe the other fundamental plan-based inference task: plan recognition.

3.2.4 The plan-recognition inference task

The input to a plan recognizer also includes a representation of the state transition system and an initial state but, instead of a goal, it takes one action (or sequence of actions) that have been observed. These elements can be formally defined as follows.

DEFINITION 9. *A* STRIPS *plan recognition problem is a triple* $P = (\Sigma, s_0, \pi)$ *where:*

- $\Sigma = (S, O, \gamma)$ *is a* STRIPS *transition system.*
- s_0*, the* ***initial state****, is a member of* S*.*
- $\pi = \langle a_1, \ldots a_k \rangle$ *is an observed sequence of actions, each action in this sequence is an instantiated instance of an operator* $o \in O$*.*

The output of this inference task is a **plan**. That is, the inference task consists in inferring the plan to which the observed sequence of action belongs to. The output plan contains all the actions that occur in the observed sequence and in the same order, but it may also include extra actions *before*, *after* or *interleaved* the observed sequence.

Intuitively it is already possible to see that the inference task of plan recognition is much harder (far more unconstrained) than planning. Let's look at Figure 3.1 and take it as our Σ again. Suppose now that the initial state is s_2 and the observed sequence of actions is $\langle \mathsf{take} \rangle$ which takes Σ into s_3. It is reasonable then to infer that the goal is to have the container loaded in the robot cart and then a possible output for the plan recognizer is $\langle \mathsf{move1}, \mathsf{take}, \mathsf{load}, \mathsf{move2} \rangle$. However, it is also possible that the crane is being tested and that the action that will follow

take is put taking Σ back to s_2. Even in this simple example, plan-recognition is already a hard task.

This inference task is so unconstrained that there are only a few symbolic methods that have been applied to it, and they work only in very restricted domains. The first one, called hypothesize and revise [Schmidt *et al.*, 1978], interleaves plan recognition and execution and consists in making a hypothesis and revising it during execution. For instance, after observing take the plan is $\langle \mathsf{move1}, \mathsf{take}, \mathsf{load}, \mathsf{move2} \rangle$ is inferred; if the next observed action is not load but say put then the plan is revised. The second method uses closed world reasoning [Kautz, 1991] which means that not only complete states are assumed (as in classical planning) but also a complete plan library. Given such a library the algorithm can infer the minimum sets of independent plans that entail the observed actions.

Given the complexity of the task, it is unsurprising that the area of plan-recognition began very early to make use of probabilistic methods: stochastic grammars [Huber *et al.*, 1994], pending sets [Goldman *et al.*, 1999], hierarchical hidden Markov models [Kelley *et al.*, 2008], among others.

3.2.5 Planning with incomplete knowledge and sensing

We can view the practical inference tasks reviewed here as organized in a hierarchy that we will call the **abduction hierarchy**. The highest tasks in the hierarchy are full abduction (as introduced in Section 3.1 and plan recognition (discussed in Section 3.2.4), with the highest computational complexities. The lowest task in the hierarchy is classical planning with the lowest computational complexity. Obviously then, complexity-wise, classical planning is the best choice. However, classical planning makes a strong simplifying assumption which impacts on its usability, namely the complete information (a.k.a. full observability) assumption.

There are different ways of dropping the complete information assumption made by classical planning. The most unconstrained scenario is one in which arbitrary literals can be added to the initial state (they can be assumed) whenever needed. If these unrestricted assumptions are allowed, the resulting inference task is again at the top of the abduction hierarchy and shares abduction's problems: many potential plans that have to be filtered out and a high computational complexity.

Planning with incomplete knowledge and sensing is an inference task that is higher than classical planning in the abduction hierarchy but lower than abduction. This variation of planning drops the complete information assumption by allowing for actions to acquire knowledge about the initial state, that is by allowing for **sensing actions** in the state transition system. The addition of sensing actions to the state transition system is not trivial, it makes the system non-deterministic as we will discuss shortly.

Planning with incomplete knowledge and sensing was inspired by how agents acting in a world acquire new information about the world. There are two sorts of sensing actions, corresponding to the two ways an agent can gather information about the world. On the one hand, a sensing action can observe the truth value of a proposition $P(c)$. We will call the

kind of incomplete knowledge sensed by this kind of action a *know whether* fact because it represents the fact that the agent knows which of the two disjuncts in $P(c) \vee \neg P(c)$ is true. On the other hand, a sensing action can identify an object that has a particular property. We will call the kind of incomplete knowledge sensed by this kind of action a *know value* fact because it represents the fact that the agent knows a witness for $\exists x.P(x)$.

There are different ways in which planning with incomplete knowledge and sensing can be implemented [Petrick, 2006]. We describe here an intuitive approach that has been implemented in a planner called PKS [Petrick and Bacchus, 2004]. PKS extends classical planning representations in order to model sensing actions. In classical planning, the world state is modeled by a single database that represents the evolving state which contains complete information. In PKS, the planner's knowledge state, rather than the world state, is represented by three databases: the *know whether* database where the *know whether* facts are stored, the *know value* database where the *know value* facts are stored, and the *know fact* database which is much like a classical planning database except that both positive and negative facts are allowed, and the closed world assumption does not apply. Actions are specified as updates to these databases.

The difference between *know fact* (henceforth K_f) and *know whether* (henceforth K_w) databases is subtle but important. So let me give you an intuitive example. Suppose I am the planner and you are the agent; as the planner, I am trying to model your knowledge state. Suppose I observe you looking outside the house (I cannot look outside myself), then I am sure that *you know whether* it is raining or not; as a consequence, raining is in the *know whether* database that I am building. Now, as far as I know, it is possible that you know that it is raining, but it's also possible that you know that it is not raining. Hence, from my point of view, your action you look outside can leave my K_f database in any of two states, either $\neg$raining $\in K_f$ or raining $\in K_f$. That is to say, and let me repeat *from my point of view*, the action you look outside is non-deterministic (see Figure 3.5); the action you look outside is a sensing act. In order to model such non-deterministic actions, K_w allows for **conditional plans** to

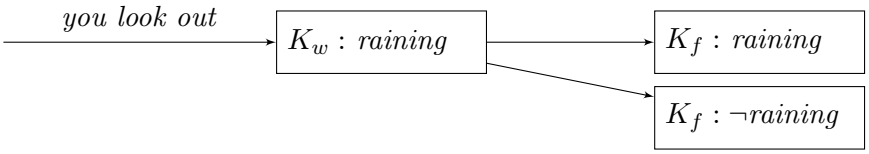


Figure 3.5: A non-deterministic action results in two states

be built (that is, plans that are not sequential but contain branches). The planner can only legitimately add a conditional branch if it is based on know-whether facts. A case-study of the use of conditional plans will be presented in Chapter 6.

Actions that modify the *know value* database result not in conditional plans but in plans with **run-time variables** [Levesque, 1996]. These plans are not ground, that is they contain a variable. A case-study of the use of plans with run-time variables will be presented in Chapter 6.

There are several levels at which knowledge can be incomplete. The most studied scenario is one in which not all the properties and relations of the objects involved in the task are known, but the set of objects is finite and all objects are named (that is all objects are associated with a constant). Intuitively this means that all objects are known when the planning problem is specified. If this simplifying assumption is made, existential and disjunctive incomplete knowledge collapse; one can be defined in terms of the other. If all objects are named, the fact that there exists an object that satisfies a particular property can be expressed as the disjunction of that property applied to all the objects in the domain. In the system we will present in Chapter 6 we cannot make this simplifying assumption because it deals with an environment where not all objects are known at plan time.

3.3 Concluding and linking the chapter

As we discussed in Section 1.2.4, an important insight about conversation is that any utterance is a kind of **action** , called a **speech act**, that is performed by the speaker. This insight is due to Austin [1962] who started to define the theory of language as part of a general theory of action, in his Speech Act Theory.

Speech Act Theory inspired Allen, Cohen and Perrault to represent speech acts as planning operators. For instance, a simplistic version of the speech act of **informing** can be specified in PDDL[5] as follows:

```
(:action inform
   :parameters (?speaker ?hearer ?prop)
   :precondition (and (believe ?speaker ?prop)
      (want ?speaker inform(?speaker ?hearer ?prop))
      (channel ?speaker ?hearer))
   :effect (believe ?hearer believe(?speaker ?prop)))
```

The preconditions of the act inform specify that the speaker has to believe the proposition, that she has to want to perform the action, and there must be a channel available between hearer and speaker. The effect of the action is that the hearer believes that the speaker believes the proposition.

Starting from this specification of speech acts as operators, models of natural language generation (NLG) were defined using the planning inference task, and models of natural language interpretation (NLU) were defined using the plan-recognition inference task. This work resulted in the influential framework for NLU and NLG called the **BDI model** (Belief, Desire, Intention model). Both directions, *generation as planning* and *interpretation as plan recognition* started a long tradition of researchers using plan-based models for natural language.

[5]The specification include literals (in the preconditions and effects) that are clearly not in first-order logic syntax but in modal-logic syntax. We use this syntax here because it makes the meaning of the action more clear; however, it is possible to recast these literals into first-order logic [Blackburn *et al.*, 2006].

Generation as planning was first proposed by Cohen and Perrault [1979]. Their model assumes that the person that starts a conversation has a goal in mind. With this goal, the speech acts as operators, and the representation of the current context as an initial state, they showed how natural language generation can be modeled using the inference task of classical planning. Perrault and Allen were the first to apply the BDI model to natural language interpretation [Allen, 1979; Allen and Perrault, 1979]. For speech act interpretation, the inference task used was plan-recognition instead of planning; intuitively, the task of the addressee is to infer the plan of the speaker. In this view, planning is done at the discourse level, the whole conversation is a plan. The speaker has a goal and a plan for achieving the goal, and all speech acts are part of the plan.

The BDI approach to NLG is still actively researched and is considered the state-of-the-art in the area. Recently, Steedman and Petrick applied planning with incomplete knowledge and sensing to this framework in order to generate in a uniform way direct and indirect speech acts [Steedman and Petrick, 2007]. The BDI approach to NLU is still the most evolved theoretical framework but has been largely replaced in practical applications (because of its computational complexity) by shallower methods [Jurafsky, 2004].

The approach adopted in this thesis is different from BDI in two essential ways. First, we use planning for interpretation. Second, plans are used to bridge the gap between an utterance and its context. That is, each utterance (and not the whole dialogue) corresponds to a plan. This approach is formally specified in Chapter 4 and implemented in a dialogue system in Chapter 6 (Chapter 5 introduces the dialogue system architecture and describe the modules that are not related to CIs).

The key insight behind our approach is that we view planning as a restricted kind of abduction for interpretation. It is a known fact that planning is a kind of abduction [Eshghi, 1988] — the actions are the theory, the goal is the observation and the initial state is the set of abducibles — in which all the assumptions have to be eventually grounded in the initial state. It can be argued that, by using planning, our framework provides a model of textual coherence, forcing the necessary assumptions to be linked to the previous context instead of allowing for the assumption of arbitrary propositions. The two paradigmatic approaches to the use of abduction for interpretation (Hobbs and Charniak) presented in Section 3.1 allow for the assumption of arbitrary propositions and then lack a model of **coherence**. Such lack was one of the main criticism of both models put forward by Norving and Wilensky [1990].

The problem with classical planning is that the model of textual coherence that it provides is too constrained. Classical planning forces all the observations to be explained to be eventually connected with the initial state through a plan. Hence, the initial state must contain all the information that is relevant for the interpretation of the observation. Otherwise, a classical planner will say "there is no plan", that is "there is no interpretation". An abductive reasoner, on the other hand, has a solution for this: if something cannot be linked, the abductive reasoner assumes it. And then abductive reasoners never say "there is no interpretation". They construct the context, they construct the initial state on the fly by assuming arbitrary propositions

whenever they are needed in order to find an explanation. But, as we have seen, this leads to too many unreasonable explanations that we invest time in computing and then we have to invest time again in discarding.

One of the main claims of this thesis is that planning extended with incomplete knowledge and sensing is a good compromise. It allows missing information to get into the state in a constrained fashion, namely only through sensing actions. To put it in another way: it allows us to go one step higher in an abstract abduction hierarchy while maintaining low computational complexity.

Chapter 4

Implicature as an *interactive* process

Therefore, as the saying goes,
it is impossible that anything should be produced
if there were nothing existing before.

from "Metaphysics," Aristotle

In this chapter, we propose a framework for inferring the CIs in an instruction and predicting its clarification potential with respect to its context; that is, we explore the CI $\rightsquigarrow$ direction of the implicature-clarification relation introduced in Chapter 2.

We start in Section 4.1 by defining the elements that constitute context in situated interaction. This is the representation of context we implement in our analysis by synthesis developed in Chapter 5. We move to a dynamic view of context in Section 4.2 where we explore how context is used in order to infer CIs. In the process, we delineate the role that practical inference plays in such a framework. Moreover, in Section 4.3, we propose this framework as the basic component of a system which can treat implicature not as one shot process but as a collaborative process, generalizing Clark and Wilkes-Gibbs [1986] treatment of referring expressions. This is the approach we implement in our analysis by synthesis developed in Chapter 6.

4.1 A static view on context

The inference framework that we present here uses four information resources whose content depends on the information available to the participants of the situated interaction we model. The interaction is asymmetric (for the reasons given in Chapter 2); that is, one of the participants (the direction giver — DG) has complete information about how the world works and the task that has to be accomplished but cannot modify the world, while the other (the direction follower — DF) can modify the world but has only partial information about the world and no information about the task. We describe each of them in turn and we illustrate their content using the SCARE experimental setup.

4.1.1 The world model

Since the interaction that this framework models is situated in a world, the first required information resource is a model of this world. The world model is a knowledge base that represents the physical state of the world. This knowledge base has complete and accurate

information about the world that is relevant for completing the task at hand. It specifies properties of particular individuals (for example, an individual can be a *button* or a *cabinet*). Relationships between individuals are also represented here (such as the relationship between an object and its location). Such a knowledge base can be thought of as a first-order model.

The content of the world model for the SCARE setup is a representation of the factual information provided to the DG before the experiment started, namely, a relational model of the map he received (see Figure A.3 in Appendix A.2.2). Crucially, such a model contains all the functions associated with the buttons in the world and the contents of the cabinets.

4.1.2 The interaction model

This knowledge base represents what the DF knows about the world in which the interaction is situated. The information learned, either through the contributions made during the dialogue or by navigating the world, are incrementally added to this knowledge base. The knowledge is represented as a relational model and this knowledge base will usually (but not necessarily) be a sub-model of the world model.

In the SCARE setup, the DF's initial instructions include almost no factual information (as you can verify looking at his instructions in Appendix A.2.1). The only factual information that he received were pictures of some objects in the world so that he is able to recognize them. Such information is relevant mainly for reference resolution and this pragmatic problem is not the focus of this thesis; only CIs are. Therefore, for our purposes, we can assume that the interaction model of the SCARE experiment starts empty.

Since this interaction model starts empty, everything that is added here is observed by both the DF and the DG, so we will assume that the information included here is mutually believed between them.

4.1.3 The world actions

The framework also includes the definitions of the actions that can be executed in the world (physical actions such as *take* or *open*). Each action is specified as a STRIPS-like operator [Fikes *et al.*, 1972] detailing its arguments, preconditions and effects. The preconditions indicate the conditions that the world must satisfy so that the action can be executed; the effects determine how the action changes the world when it is executed.

In SCARE, these actions specify complete and accurate information about how the world behaves, and, together with the world model, they are assumed to represent what the DG knows about the world. The SCARE world action database will contain a representation of the specification of the QUAKE controls (see Appendix A) received by both participants and the extra action information that the DG received. First, he received a specification of the action *hide* that was not received by the DF. Second, if the DG read the instructions carefully, he knows that pressing a button can also cause things to move. The representation of this last action schema is shown in Appendix A.2.2.

4.1.4 The potential actions

The potential actions include a definition of how the DF conceptualizes the actions that he can perform on the world. This knowledge base may (but need not) coincide with the world actions. If it does not coincide it means that the DF has misconceptions about some actions.

In SCARE, the potential actions include representation of actions that the DF learned from the instructions he received before beginning the task. This includes the QUAKE controls (see Appendix A) and also the action knowledge that he acquired during his learning phase (see appendix A.2.1). In the learning phase the direction follower learned that the effect of pressing a button can open a cabinet (if it was closed) or close it (if it was opened). Such knowledge is represented as a STRIPS-like operator like one showed in Appendix A.2.1.

4.1.5 Going dynamic with practical reasoning

In this section, we have specified the different elements that constitute the context of the SCARE experiments. These elements play a crucial role in the inference of the CIs: In order to infer the CIs of an utterance it is crucial to understand the dynamics of these elements. That is, in order to infer the CIs of an utterance it is crucial to understand the dynamics of its context.

A state of a dynamical system is usually specified by certain dynamical variables. If the number of such variables is very large the methods of analysis is likely to be statistical in nature, and consequently the predictions made shall be the most probable ones. If the number is not large, then the system is amenable to symbolic analysis. We consider this last to be the case for the restricted setup of the SCARE experiments, so our model will be symbolic.

In this section we discussed the context variables that are necessary in order to come up with the clarification potential of an instruction, namely the interaction model in Section 4.1.2 and the world model in Section 4.1.1. We also presented here two elements of the context that we assume to be static, namely the potential actions in Section 4.1.4 and the world actions in Section 4.1.4 (assuming that the potential actions are static is a simplification because the DF's understanding of the world actions can change during the interaction). The actions represent how the context can evolve from one state to the next; that is, they represent the causal links of the SCARE game world.

4.2 Computing with context

In the next three subsections that follow we present a framework that spells out how implicated premises, implicated conclusions and explicatures are inferred and used in conversation. The framework design is based on the empirical evidence we found in the SCARE corpus (see Chapter 2).

A generic inference framework for CIs

The inference framework that we present here links the CIs of an utterance with its CRs through the following four steps:

Step 1: Pick an instruction from the corpus.
Step 2: Calculate the CIs of the instruction.
Step 3: Predict the CRs using the CIs.
Step 4: Compare the predictions with the corpus.

In the next section we are going to apply this procedure to the three kinds of CIs distinguished by relevance theory: implicated premises (Section 4.2.1), implicated conclusions (Section 4.2.2) and explicatures (Section 4.2.3). We close this section with an evaluation of the coverage that this framework has in the SCARE corpus.

4.2.1 An inference framework for implicated premises

In this section we do two things. First, we say how current off-the-shelf planners can be used to infer the implicated premises of an utterance with respect to its context. Second, we explain the cases when implicated premises are made explicit in clarification requests.

Let's do these two things by further specifying steps 2 and 3 of the generic framework, proposed in the previous section, for the case of implicated premises.

Step 1: Pick an instruction from the corpus.

Step 2: When the DG gives an instruction, the DF has to interpret it in order to know what actions he has to perform. The interpretation consists in trying to construct a plan that links the current state of the interaction with the preconditions of the instruction.

An action language (like those introduced in Chapter 3) can be used to specify the two action databases introduced above. The world model and the interactional model are relational structures like the one showed in Figure A.3 (in the appendix). These relational structures can be directly expressed as a set of literals which is the format used to specify the initial state of a planning problem. These information resources then constitute almost everything that is needed in order to specify a complete **planning problem**, as expected by current planners: the only element that the framework is missing is the goal. Remember from Chapter 3 that with a set of action schemes (i.e. action operators), an initial state and a goal as input, a planner is able to return a sequence of actions (i.e. a plan) that, when executed in the initial state, achieves the goal.

In short, the specification of such planning problem is as follows.

- The preconditions of the instruction are the **goal**.
- The dialogue model is the **initial state**.
- The potential actions are the **action operators**.

With this information an off-the-shelf planner finds a sequence of actions (that is, a plan), or says there is no plan. This sequence of (tacit) actions is the set of implicated premises of the instruction.

Step 3 (if there is a plan): After inferring the plan an attempt to execute the plan on the the world model and using the world actions occurs. Whenever the plan fails, the tacit act that failed is a potential CR.

Step 3 (if there is no plan): All the preconditions that cannot be linked to the initial state by a plan are added to the set of potential CRs.

Step 3 (if there is more than one plan): The plans are ranked in some way and the tacit acts of the higher ranked plan are part of the clarification potential of the instruction.

Step 4: Compare the predictions with the corpus.

In summary, the **implicated premises** of an instruction is the sequence of (tacit) actions that links the previous interaction model with the preconditions of the instruction. This framework gives three possible scenarios: there is a sequence which fails (failed plan), there is no sequence (no plan), there is more than one possible sequence (multiple plans). We will illustrate each of them in turn as follows.

The plan fails

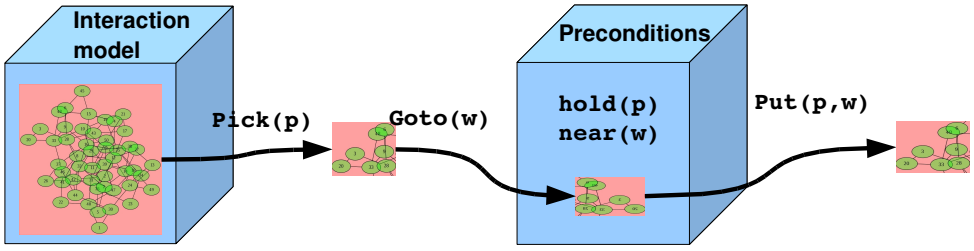
Let's now instantiate the framework proposed in the previous section to analyze an example of implicated premises from the SCARE corpus. In this example the participants are trying to move a picture from a wall to another wall (task 1 in Appendix A.2.2). Let's go step by step:

Step 1: The instruction that is being interpreted is DG(1).

DG(1): well, put it on the opposite wall

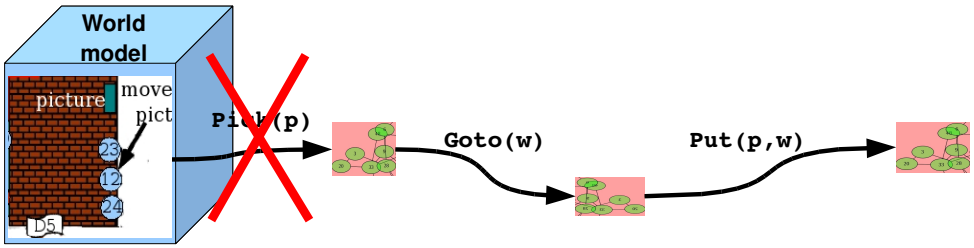
Step 2: The preconditions of this instructions are to have the picture (in the DG's hands) and to be near the target wall.

The inference process is illustrated in the figure below.



The inferred plan involves *picking up the picture* in order to achieve the precondition of *having the picture*, and *going to the wall* in order to achieve the precondition of *being near the wall*. That is, `Pick(p)` and `Go-to(w)` are CIs of what the DG said.

Step 3: The plan inferred by the DF and showed in the figure above fails in the game world because the picture is *not takeable* and thus it cannot be picked, resulting in a potential clarification. The action that fails is illustrated in the following figure with a red cross.



The correct plan to achieve (1) involves pressing button 12, as you (and the DG) can verify on the map (in the Appendix).

Step 4: In the corpus, the potential clarification, foreshadowed by (2) and (3), is finally made explicit by the CR in (4), as predicted by the model.[1]

DF(2): ok, control picks the .
DF(3): control's supposed to pick things up and .
DF(4): am I supposed to pick this thing?

There is no plan

In the case that no plan can be inferred, our framework predicts that the instruction precondition for which no plan can be found will be part of the clarification potential of the instruction. Consider the following example.

Step 1: In the dialogue below, the DG utters the instruction (1) knowing that the DF will not be able to follow it; the DG is just thinking aloud.

DG(1): we have to put it in cabinet nine .

Step 2: If taken seriously, this instruction would have the precondition *the reference "cabinet nine" resolved*. However, this precondition cannot be achieved by any action because the DF doesn't know the numbers of the cabinets. Hence, a planner can find no plan for this planning problem.

Step 3: The framework then predicts a CR related to resolving the referent "cabinet nine".

Step4: Both participants know that the DF does not know the numbers, only the DG can see the map. That's why the CR in (2) is received with laughs and the DG continues his loud thinking in (3) while looking at the map.

DF(2): yeah, they're not numbered [laughs]
DG(3): [laughs] where is cabinet nine .

[1] In the dialogue fragments, the '.' indicates a pause.

Our framework would not be able to produce a clarification move as precise as the DG did above because the planner will just say there is no plan for resolving the reference 'cabinet nine'. However, using the information that the framework can output, namely "the referent 'cabinet nine' cannot be resolved", a more general clarification such as "What do you mean by cabinet nine?" can be produced.

The plan is uncertain

In the case that more than one plan can be inferred for the given instruction, the alternative plans will be part of the clarification potential of the instruction. Why? Because the DF cannot be certain which was the plan that the DG had in mind. We can see the following dialogue (which continues the fragment above) as an instance of this case.

Step 1: Now, the DG refines the instruction given in (1) with the location of the target of the action put in (4).

DG(4): it's . kinda like back where you started . so

Step 2: And the DF comes up with a plan that achieves the precondition of the instruction *put* of being near the destination of the action (cabinet nine) in this case being *where you started*.

Step 3: Uttering the steps of the plan that were not made explicit by the instruction is indeed a frequently used method for performing clarification of hypotheses. The DF clarifies hypotheses when he is not certain that the plan he found is exactly what the DG wants.

Step 4: The DF incrementally grounds the first plan he found by making it explicit in (5), (7), and (9) and waits for confirmation before executing each action.

DF(5): ok . so I have to go back through here?
DG(6): yeah
DF(7): and around the corner?
DG(8): right
DF(9): and then do I have to go back up the steps?
DG(10): yeah
DF(11): alright, this is where we started

The DF clarifies hypotheses when he is not certain that the plan he found is exactly what the DG wants. An obvious case of that is when there is more than one plan. However there might be other sources of uncertainty: for instance, the DF's memory is not perfect. We discuss such cases in Section 4.2.4.

4.2.2 An inference framework for implicated conclusions

Not only the implicated premises of an instruction can be clarified but also its implicated conclusions. However, the implicated conclusions cannot be inferred using the method that we presented in the previous section. This is because the method infers the actions that are *necessary* in order to execute the instruction and the implicated conclusions are not necessary, but only *normally* drawn from the instruction and the context. In this section, we use a practical inference task that is different from classical artificial intelligence planning.

Step 1: Consider the following example, the DG told the DF to press a button, in turn (1) and (2), with no further explanation. As a result of pressing the button a cabinet opened.

DG(1): now, on the wall on the right turn and face that
DG(2): press the button on the left
DF(3): [presses the button and a cabinet opens]

Step 2: The inference of implicated conclusions can be defined intuitively as a practical inference task which involves finding the set of **next relevant actions**. The input of this means-ends task is different to that of a planning problem. It also consists of an initial state, and a set of possible actions, but it will also contain one observed action (in the example, action (4)) instead of the goal. Inferring the next relevant action consists in inferring the affordabilities (i.e. the set of executable actions) of the initial state and the affordabilities of the state after the observed action was executed. The output of this inference task, the set of next relevant actions are those actions that were activated by the observed action. In the example, the next relevant action that will be inferred is "put the thing you are carrying in the cabinet that just opened".

Step 3: As far as we observed, in the SCARE corpus the DF never executes a next relevant act without clarifying it beforehand. Such acts are possible follow ups but they are not certain. This is probably a result of the setup of the experiment which punishes the dialogue participants (DPs) if they perform wrong actions.

Step 4: In the example above, the next relevant action that will be inferred is "put the thing you are carrying in the cabinet that just opened", just what the DF verbalizes in (4). In (5) the DG confirms this hypothesis.

DF(4): put it in this cabinet?
DG(5): put it in that cabinet, yeah

4.2.3 An inference framework for explicatures

Once the size of the unit of update of the elements of the context was fixed, as happened when we defined the actions for the SCARE experiment in Section 4.1, the line between explicatures

and implicatures was fixed for our framework. It was suggested to me[2] that we could have used a bigger size for the unit of update of my context (that is, my actions) and then the example about *putting the picture on the opposite wall* from Section 4.2.1 could be treated as a missing manner parameter (and hence be covered in his formalization of intended content CRs, see Chapter 2). On this view, the semantic content of the utterance *putting the picture on the opposite wall* has a *manner* missing parameter that can be filled by *by picking it up*. In this way, we would avoid all the "heavy reasoning on plans and intentions". Certainly this can be done for this example, however we don't think that the heavy reasoning can be avoided for long. If we increase the size of our unit of context update every time we find such a CR with an extra argument we would end up with, in principle, an infinite number of additional arguments (consider for instance the dialogue discussed in Section 2.3, where it takes 25 turns to finally ground the instruction *put the rebreather in cabinet 9*, and this is not, it must be emphasized, a rare example). That is, we would have a pragmatic analog of the semantic problem that Davidson solved many decades ago when he proposed his event semantics [Davidson, 1967].

So not all CRs can be explicatures.[3] And the line between implicatures and explicatures has to be fixed. As follows we present CRs that, in our framework, fall under the label of explicatures.

Taking this point of view, we encountered in the SCARE corpus instances of explicatures that we interpret in our framework as parameters of the task action that are missing and cannot be inferred from the context. For instance, in the following exchange the DG gives the instruction *take the stairs* and the DF does not know whether he should go downstairs or upstairs. These are the two possible values that the missing parameter of this action can take and the DF clarifies in (2) which is the intended one.

DG(1): there should be some stairs . take the stairs .
DF(2): up? .
DG(3): yeah

Now, there is evidence in the corpus that the DF expect the DG not to provide parameters of actions that can be inferred from the context. For instance, in the following dialogue, the DG specifies which way to leave the room in (1). However, this is the only exit of the room in which the DF is currently located and the DF makes this explicit in (2).

DG(1): Let's leave the way [pause] we came
DF(2): that's the only way .

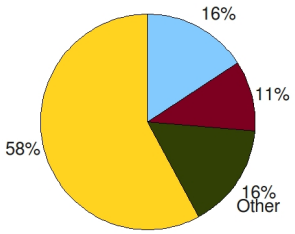
[2] p.c. with Jonathan Ginzburg.

[3] We think that the other direction is more interesting: probably all explicatures can be treated as implicatures. This would require a smaller size for the context update than task-meaningful physical actions. It would imply that the interactions between the explicatures of an utterance would not need to be revised in the light of the implicatures, rather, all the not-explicit content would be developed in parallel embedded within the overall process of constructing a hypothesis about the speaker meaning. Whether such a framework can be properly formalized is a task for further work.

In a broader framework than the one that we present here, we envisage the task of finding the explicatures of an utterance to the task of identifying the **speech act** made with that utterance as defined by Jurafsky and Martin in [Jurafsky, 2004] (see Section 1.2.4). However, we don't agree with the theory of speech acts that a general taxonomy of speech acts can be found for dialogue (such as the DAMSL –Dialogue Act Markup in Several Layers [Allen and Core, 1997]). Rather, we think that this taxonomy has to be defined for each task and that the process of finding the explicatures is the process of mapping surface form of a contribution to its task move where all the face preserving strategies have been removed: namely when the politeness strategy called **bald on record** is used (see Section 1.2.2). Such framework makes sense if task success is the only goal of the interaction and then it is only suitable for task-oriented interaction. Frameworks that model interactions in which, for instance, social goals are relevant, would not want to throw away the politeness layers. Whether social goals and strategies can be formalized in a task-structured way remains a task for for further research.

4.2.4 Discussion and evaluation of the frameworks

We used the classification of conversational contributions into CRs presented in Chapter 2 in order to spot 4th level CRs in the SCARE corpus. And then we applied the frameworks described in this section in order to classify those CRs into:



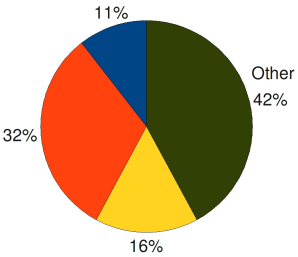
Implicated premises (depicted in yellow).

Implicated conclusions (depicted in light blue).

Explicatures (depicted in dark red).

The classification is done according to the inference task that is used in order to infer them. In the SCARE corpus this gave us a 85% coverage. That is, we were able to calculate 85% of the CRs that appear in the portion of the SCARE corpus that we analyzed. The percentage of each kind of CR is shown in the figure. Most of them correspond to implicated premises.

We further classified implicated premises according to the reason why the CI had to be made explicit in a CR. The reasons for classifying an implicated premise are listed below and the percentages found on the corpus are shown in the pie chart.



Wrong plan: the plan inferred is not executable (depicted in yellow).

Not explainable: no plan can be inferred (depicted in blue).

Ambiguous: more than one plan can be inferred (depicted in orange).

The framework we presented here can be used in two directions, offering two different results:

CRs $\rightsquigarrow$ CIs A quantitative way to evaluate systems that infer conversational implicatures: A system based on classical planning can infer 58% of the implicatures made explicit in the SCARE corpus; that is, it can infer all its implicated premises.

CIs $\rightsquigarrow$ CRs It offers the first framework to classify clarifications (in a fine-grained fashion at level 4) according to their associated inference task. It gets a 84% coverage in the portion of the SCARE corpus we analyzed.

The CRs not covered by the classification (16% in the SCARE corpus) have to do mainly with the fact that people do not completely remember (or trust) the instructions given for the experiments or what they (or their partner) said a few turns before. Here is an example:

DG(1): you've to . like jump on it or something .
DF(2): I don't know if I can jump

Here, the DF does not remember that he can jump using the space bar as stated in the instructions he received (Appendix A).

In order to account for these cases it is necessary to consider how conversation is useful for overcoming this issue. The fact that people's memory is not reliable is intrinsic to communication and here again, communication must provide mechanisms to deal with it. Modeling such things are challenges that a complete theory of communication will have to face.

However, with the results that we already have we can go back to the intuitions from Chapter 1 and argue that the framework explains how conversational participants do granularity switching. On the one hand, CRs associated with implicated premises are an intrinsic mechanism of conversation that allows the dialogue participants to switch to a lower level of granularity.[4] On the other hand, questions associated with implicated conclusions (which turn out to be correct next moves) suggest that the DF knows the structure of the task well enough to make correct predictions. Such moves by the DF might give the DG the feeling that the DF is moving ahead faster than him and thus be a good indication that the DG needs to move to a higher granularity. These intuitions set the bases for starting to delineate conversational implicatures not as something that the speaker does but as the result of a collaborative negotiation between the dialogue participants. We will present the sketches of such a model of conversation in Section 4.3.

4.3 Implicature as an interactive process

Clark and Wilkes-Gibbs [1986] have demonstrated that participants in conversation work together in the process of making of a definite reference. More generally, they argue that the

[4]There is empirical evidence that shows that low-level CRs result in the dialogue staying in a lower level of granularity in further exchanges [Mills, 2007]

participants in a conversation try to establish the mutual belief that the listeners have understood what the speaker meant in the last utterance to a criterion sufficient for current purposes.

With definite reference, the participants's attempts take the form of an acceptance process. The speaker initiates the process by presenting or inviting a noun phrase. Both speaker and addressees may repair, expand on, or replace this noun phrase in iterative fashion until they arrive at a version they mutually accept. In this process they try to minimize collaborative effort, presenting and refashioning these noun phrases as efficiently as possible. One result is that the preferred form of reference is the one in which the speaker presents an elementary noun phrase and the addressees presuppose their acceptance of it without taking an extra turn.

Although their work focuses on reference resolution, Clark and Wilkes-Gibbs are aware that this general process that they describe may also describe other parts of the speakers meanings:

> *Participants in a conversation, we have argued, are mutually responsible for establishing what the speaker meant. Definite reference is only one part of that process. They must collaborate, in one way or another, on most or perhaps all other parts of speakers meaning as well. [Clark and Wilkes-Gibbs, 1986, p. 37]*

An implicit argument behind this thesis, which goes all the way back to Chapter 1, is that participants in a conversation must *collaborate*, or as we prefer to say, they must *interactively negotiate* on the implicatures that are being made in a conversation.

4.3.1 How does the process start?

We have analyzed in this chapter the implicatures of instructions, so for the process to start it is necessary that the speaker gives an instruction and that the hearer tries to link it to the current conversational context. If the link cannot be directly constructed because the preconditions required by the instruction are not satisfied in the conversational context then the interactive process of implicating starts. However, our framework does not only apply to instructions. Let's go back to Grice.

Applying our framework to Grice's relevance implicatures

Let's return to the example by Grice that we first discussed in Chapter 1 and analyze it in the light of the inference framework developed in this Chapter. We reproduce the example here:

(9) *A: I am out of petrol.*
B: There is a garage around the corner.
B conversationally implicates: the garage is open and has petrol to sell.
[Grice, 1975, p.311]

So the first task is to calculate the explicatures (or assertional implicatures in Thomason's terms [Thomason, 1990]) of the two utterances of this dialogue. As we argued in the last

section, in our framework this amounts to removing the politeness layers and coming up with the bald on record contribution that has been made.

Let's think about A's utterance. A utters *I'm out of petrol* standing by his car. It can be argued that A is using an off-the-record strategy here (see Section 1.2.2 for a definition of the different politeness strategies). According to politeness theory, many cases of off-record acts are accomplished by stating some undesired state of affairs (such as "it's cold in here") which are a reason for some desired act A (such as "shut the window"). The state of affairs might suggest more than one possible desired act (such as "turn on the heating") and it's up to the hearer to choose which one to perform, if any. The choice of such a high level politeness strategy in this exchange is justified by the fact that A and B do not know each other; B is just a passer by. B might have interpreted A's utterance as also meaning "give me petrol" but he didn't, as made evident by his reply. We say that the explicature he decided to take up is the following:

(10) *A: I'm out of petrol))))) A: Tell me where to get petrol*

Since B's contribution constitutes an answer to A's bald on record request "Tell me where to get petrol" then its bald on record contribution is the following.

(11) *B: There is a garage around the corner))))) B: Get petrol in a garage around the corner*

Now, we are ready to calculate the implicated premises of B's contribution. The preconditions of the action *A gets petrol in X* that we represent are four, as depicted in Figure 4.1.

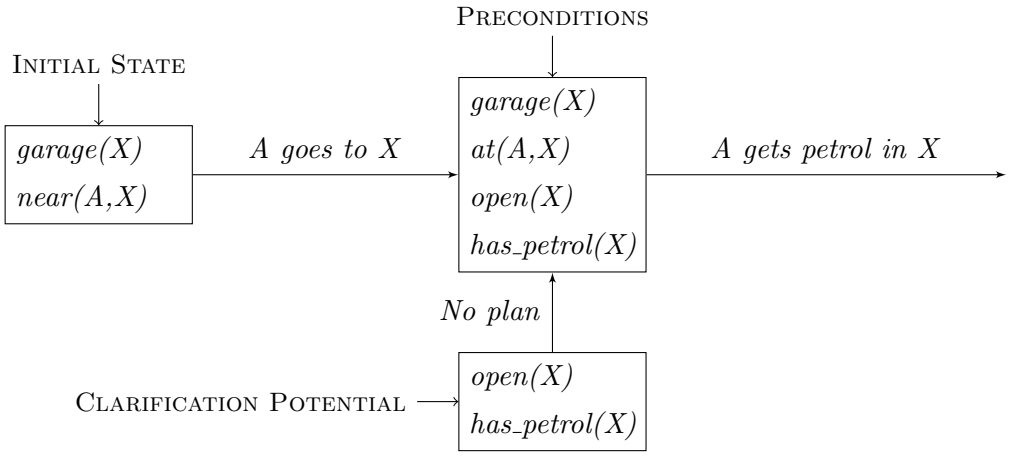


Figure 4.1: Inferring the implicated premises of Grice's example (9)

The first precondition, namely *garage(X)* (which means that X is a garage) holds in the initial state, thus is already achieved and no tacit act is necessary. The second precondition, *at(A,X)* which means that A is at the garage, can be achieved with the action *A goes to X* (whose precondition is *near(A,X)*). However, there is no plan for achieving the third and fourth preconditions, namely *open(X)* and *has_petrol(X)*.

So the prediction of the framework is that the clarification potential of the sentence will include *open(X)* and *has_petrol(X)* and will not include *at(A,X)* because it can be achieved by

a tacit act which is executable (A can go to the garage). As a consequence it will be coherent for A to continue the exchange with either (12a) and (12b) but not with (12c).

(12) a. *A: and you think it's open?*

b. *A: and you think they have petrol?*

c. *A: and you think I am at the garage?*

So if we came back to Grice's terminology, we can say that the fact that the garage is open and has petrol to sell has been implicated by B. We shall argue, though, that our framework also predicts that the fact that A will have to go to the garage is also implicated by B. And we think that this prediction makes sense because it wouldn't be collaborative for B to say *there is a garage around the corner* if it is obvious that A cannot go around the corner (imagine that A is in a wheelchair and there are big stairs leading to the corner).

Generalizing exploitation

In Grice's garage example, the person who doesn't have petrol (namely A) comes up with the CIs "the garage is open" and "has petrol to sell" when attempting to construct the link to the current context. Once A has inferred that these two CIs, he might accept these inferences silently or negotiate them, for instance in a clarification request such as "and you think it's open?" and "and you think they have petrol?" respectively. A might decide to silently accept them because he thinks that garages are usually open at that time of the day and they that almost always have petrol to sell. Or A might decide to accept it because B is his boss and A knows that B only make sensible recommendations and does not like 'silly' questions. In these cases we will say that A constructed an **internal bridge** from the clarification potential to the initial state. If A decides he has not enough evidence in order to construct the internal bridge, A starts a sub-dialogue that we will call **external bridge**. The underlying hypothesis of this thesis is that by observing those external bridges (those sub-dialogues, those clarifications) we will learn how people construct their internal bridges.

Whether the hearer decides to construct an internal bridge or an external one depends on how plausible seems the internal bridge to be. We could argue that A will probably decide to query "and you think it's open?" instead of querying "and you think they have petrol?" because it is more probable that a garage is closed than that it run out of petrol.

In general, the process starts because the speaker makes a contribution that is exploiting (a.k.a *flouting* in Grice terminology) some convention. Wherever some convention or expectation about the use of language arises, there will also therewith arise the possibility of the **exploitation** of that rule or expectation. In our view, exploitation is a phenomena that is more general than linguistic communication and can be observed in activities that do not involve language at all. In [Thomason *et al.*, 2006], the authors present a completely physical example which is worth reproducing.

> *A woman at a restaurant violates the rule (R1) "Don't sit down unless there is an empty chair positioned behind you," knowing that a waiter will position the required chair, and that indeed her sitting will be the signal for this act. [Thomason et al., 2006, p. 36]*

In this example, the woman *exploited* the convention (R1) to *mean* that the waiter should position the required chair. If the waiter reacts in the expected way we can say that the waiter *accommodated* the wishes of the woman.

In interaction that interleaves physical and linguistic acts, the rules to the actions involved in that activity also have conventional force and can be exploited when using language. The **felicity conditions** [Levinson, 1983] of these rules are its preconditions. In this thesis we concentrate on how rules, which are associated to the actions of the task in which the conversation is situated, are exploited. This is one step in a direction that we think it's worth pursuing: we think that the program of extending exploitation to all appropriateness conditions (at different linguistic and interactional levels) would result in a general basis for pragmatic theory.

Summing up, when the hearer is trying to link the speakers contribution to the context and he recognizes that exploitation is being used then he has one of two options:

- He starts an internal process of bridging.
- He starts an external process of bridging.

The internal process of bridging is what in the literature has been called accommodation or bridging. The external processes of bridging constitutes a large part of what we call conversation.

4.3.2 The internal bridging process

In 1979, Lewis coined the term **accommodation** as a label for the following process, which he viewed as governing conversation:

> ***Accommodation:*** *If at time t something is said that requires component s of conversational score to have a value in the range r if what is said is to be true, or otherwise acceptable; and if s does not have a value in the range r just before t; and if such-and-such further conditions hold; then at t the score-component s, takes some value in the range r. [Lewis, 1979, p. 347]*

Lewis argued that this is the process by which hearers *fill in a gap* left by the speaker. The gap occurs because something is missing from the context of the conversation and is required by what the speaker said. For an up-to-date review of accommodations of presuppositions see [Beaver and Zeevat, 2007].

A few years earlier, Clark coined the term **bridging** in order to label the process by which hearers *fill in a gap* left by the speaker. In his view, bridging is a consequence of a speaker-listener agreement known as the Given-New contract.

> ***Given-new contract:*** *The speaker agrees to try to construct the Given and New information of each utterance in context (a) so that the listener is able to compute from memory the unique antecedent that was intended for the Given information, and (b) so that he will not already have the New information attached to the antecedent. [Clark, 1975, p. 170]*

In the simplest case, the strategy just given will work without problems. In the more typical case, however, the speaker will not follow this rule but exploit it. When this happens, the listener won't find such an antecedent directly in memory. He will be forced to construct an antecedent, by a series of inferences, from something he already knows. The construction of such an antecedent by performing inferences on what he already knows is the process called bridging.

The distinction between accommodation and bridging is explained by Beaver and Geurts [in press] with the following example by Heim [1982].

(13) *John read a book about Schubert and wrote to the author.*

They analyze the example (13) in an intuitive way, saying that in order to determine the intended meaning of "the author", the hearer has to infer (i) that there is an author and (ii) that the said author wrote the book read by John. But then they make explicit a simplification that is usually left unsaid in most of the literature on accommodation.

Most of the literature on accommodation assumes a strict notion of accommodation and does not take into account (ii) when analyzing example (13). We agree with Beaver and Geurts that the difference between accommodation and bridging is not terminological; in fact it is a crucial distinction:

> *Whereas on a broad understanding of accommodation, all of this is accommodated, on a strict construal only (i) is, and (ii) is a bridging inference. This is not just a matter of terminology. If we choose to be strict, we can argue that there is something like an "accommodation module", which as such has nothing to do with world knowledge; whereas if the notion is construed more broadly, accommodation is of a piece with bridging. [Beaver and Geurts, in press]*

The difference then between accommodation and bridging is then that in accommodation, the required antecedent (e.g. the antecedent for the author) is just added to the context while in bridging the antecedent has to be constructed *from something the addressee already knows* (e.g. there is a salient book in the context and books have authors) by a series of inferences.

Beaver and Geurts explicitly decide to adopt a strict notion of accommodation, because "it facilitates the following discussion". However, we think that accommodation theory should consider how the bridges into the conversational context are constructed (either internally or externally) in order to throw new light on many of its current puzzles. In what follows we discuss some of them.

Why isn't it always equally easy (or hard) to accommodate?

Van der Sandt [1992] observes that presuppositions whose descriptive content are relatively poor are hard to accommodate. For instance, if a speaker A uttered (14) out of the blue and the context did not contain a salient female person, the addressee B would be confused.

(14) *A: She is beautiful.*
B: Who?

The problem with Van der Sandt's comment is that, even if it somehow makes sense that lack of descriptive content will make accommodation hard, it is not clear why this should be so, or even whether this is always the case (for instance, is the descriptive content of "the happy bride" relatively poor?). B&G argue that a possible answer to this question may be found in work on definites such as [Clark and Marshall, 1981; Clark, 1996]. Clark argues that in order to make a successful definite reference the referent must be inferable from what is jointly salient in the speaker's and addressee's common ground at that moment. B&G then conclude that accommodation is easier with richer presuppositions, not because they have more content, but simply because they are more likely to contain anchors into the common ground.

We think that this answer is on the right track. However, some comments are in order. First, certainly for the case of definite references (and probably for accommodation in general) it is not enough that the presuppositions contain anchors into the common ground. Suppose, for example, that I tell my sister "I just met the person that someone saw" there will be *many* potential referents in our common ground that would fit the definite description "the person that someone saw" and then the definite reference will fail. That is, for definite descriptions to be successful the anchor in the common ground has to be unique. Second, although not made explicit in the B&G argument, the use of the common ground for solving this question results in broadening the concept of accommodation and falling again into bridging; and then, this question cannot be answered in a theory of strict accommodation. The solution to this problem in bridging terms would be: if a bridge to the common ground can be constructed, and for definite reference is unique, then it's easy to accommodate, otherwise it will be hard. we want to add here, if accommodation is hard, that is, if internal bridging fails then, normally (unless the matter is dropped), external bridging starts (e.g. my sister will ask "what do you mean by the person that someone saw?").

What happens when a presupposition is false?

The problem of presupposition failure addresses the question: what happens if a presupposition is false (wrt the world)? Is the utterance false as a result or does it lack a truth value? This problem is the oldest one in the theory of presupposition and many researchers lost interest in it (and concentrated on problems such as presupposition projection [Geurts, 1999; Schlenker, 2007]). Strawson [1964] argument on the importance of this issue is one of the few arguments that still survive nowadays [Beaver and Geurts, in press]. In this paper, Strawson does not

commit himself to any of the standard answers to the question raised by this problem (the utterance is false or it lacks a truth value), but makes the problem relative to topic-focus issues.

If we take a common ground approach to this problem, as B&G did for the previous problem, the formulation of presupposition failure goes along the following lines: what happens if a presupposition is false with respect to the (assumed) common ground? The crucial difference between this approach to the problem and the traditional one is that the information in the (assumed) common ground might be *incomplete* and *incorrect*. Since conversational partners are aware of this they might rather question a false presupposition instead of rejecting it. For instance, if the speaker is supposed to know more about the truth of the presupposition than the addressee (for instance is A is living in France and B is living in South America) the utterance might not be rejected but questioned:

(15) *A: The king of France is bald.*
B: Does France have a king?.[5] *I thought they didn't.*

In this case, we would say that B started an external process of bridging. A and B can then negotiate the truth value of the proposition "France has a king". Whether this proposition is actually true of false in the world is an independent matter and it's not necessary to determine it in order to obtain a internally coherent conversation.[6]

In the case that the speaker A is known to have correct information about the exchange the addressee might even accommodate a false presupposition in the following way.

(16) [B thinks that "Mary is not married" is in the common ground between Mary's sister and B but haven't seen Mary or Mary's sister for a couple of years.]
Mary's sister: I have to pick up Mary's husband at the airport.
[B updates the common ground with "Mary got married" which removes "Mary is not married" and adds "Mary has a husband"]
B: When is he arriving?

In this case, we would say that B did an internal process of bridging executing the tacit act of "Mary got married" on the common ground.

In this view of the problem, an utterance with a false presupposition starts an accommodation process similar to the standard accommodation process that is started by a missing

[5]Strange as it may sound (specially to semanticists) this question is frequent enough to be included in http://answers.yahoo.com/ with some sensible answers (*No, they have a republic*), some funny ones (*They have Burger King but they don't have French fries*) and some surprising ones (*Yes, only a "pretender" King. A "pretender" is a claimant to an abolished throne or to a throne already occupied by somebody else. There are currently three "pretenders" to the French throne: [...]*). If these pretenders succeed, semanticists will need to change their favorite example.

[6]Of course, interlocutors are constantly striving to have a common ground that is consistent with the actual world (having wrong beliefs is bad because they lead to wrong predictions, remember Chapter 1) and then felicitousness and truth seem to be inevitably linked nonetheless, they might be better studied independently.

presupposition. As we said above, this is possible because the common ground may contain incorrect information that can be revised. Of course, if after the bridging process the utterance cannot the linked to the common ground (for instance, the speaker is not able to support the claim) then the utterance might be rejected.

We view this setup as a generalization of the accommodation problem: not absent presuppositions but also false ones can be bridged (if it is possible to build a bridge). In this setup an utterance with a false presupposition in not viewed as an utterance that lacks a truth value (or that is false) but as an utterance that starts a bridging process which requires tacit acts. The prototype that we present in Chapter 6 is a proof of concept of how this generalized approach can be implemented in a conversational agent.

Interestingly, in this common ground approach, the observations made in Strawson [1964] on how topicality relates to presupposition failure can be explained along the following lines. Intuitively, an utterance will be connected to the common ground through its topic, so if a presupposition affects the topic of an utterance the utterance is infelicitous (or lacks a truth value) because the utterance cannot be connected to the common ground. On the other hand, if the presupposition failure does not affect the topic then the utterance can still be connected to the common ground, hence it is less likely for that utterance to be considered infelicitous. This kind of analysis of Strawson's observations seems promising if topicality of an utterance was an easy concept to pin down. In reality, understanding the topic-focus distinction is far beyond the scope of this thesis, but is is certainly a path worth pursuing.

Where do we accommodate?

The last problem we review here under the light of common ground is the largely unexplained observation of accommodation theory. Namely, the principle of global accommodation (PGA): Global accommodation is preferred to non-global accommodation. In accommodation theory it is not known why the PGA should hold.

However, if we move to a bridging view of the phenomena we can argue that the presupposition will be accommodated into the context in which it can be bridged to. For instance, suppose that we are organizing the dinner of a conference and we need to know the maximum number of people that may attend, and I tell you:

(17) If the invited speaker is coming to the dinner, he may bring his wife.

An overhearer might be tempted to assume then that I know that the invited speaker is married. However, you know (because you are my co-chair) that we still do not know who our invited speaker is going to be; so I cannot possibly know whether he is married or not. No inconsistency would result from adding "the invited speaker is married" to our common ground (and apply the PGA principle) but the fact is that we do not have enough information in order to construct this bridge, we do not have a referent for the invited speaker in our common ground, at least not one to which we can attach a wife. And then you know that what I really meant with my utterance was:

(18) If the invited speaker is married and is coming to the dinner, he may bring his wife.

If accommodation theory stops analyzing language as an overhearer and tries to model the accommodation that happens routinely in everyday dialogue where interlocutors are using language for actually doing things, the theory will probably have to revise some of its long lasting principles.

4.3.3 The external bridging process

During a dialogue the three sources of external bridging (that we presented in Section 4.2) interact resulting in the emergent structure that we can then appreciate when we analyze a dialogue in *retrospect*. The following example shows how the *no plan* case can interact with *uncertain plan*.

The continuation of the dialogue above is an instance of using implicated premises when there is no plan. In (13) the DF is not able to find a plan that achieves another precondition of the action *put*, namely that the destination container is opened, so he directly produces a CR about the precondition.

DG(12): ok . so your left ca- . the left one
DF(13): alright, so how do I open it?

However, the DF does not stop in (13) and wait for an answer but he continues with (14) as you can see below.

DF(14): one of the buttons?
DG(15): yeah, it's the left one

Here, a classical planner will just say "there is no plan" explaining how (13) but not (14) are generated. These are cases in which no complete plan can be found but the DF is anyway able to predict a possible course of action. The information necessary to produce (14) cannot be obtained by a classical planner but then non-classical planners that find plans when information is incomplete (which are introduced in Chapter 3) are a solution here.

The following example illustrates how uncertain plans and failed plans can interact. The DF comes up with two plans that are supposed to achieve the instruction given in (1) and (2). One plan involves "pressing control" and the other sequence involves "jumping on the object". However, the DF is being trained (in the learning phase) to pick up objects pressing Ctrl so he silently tries this plan first and he verbalizes in (3) the second plan, that is the dis-preferred plan.

DG(1): we wanna move those three boxes
DG(2): so they're all three on the left side of the table
DF(3): ok may be I try jumping in and up there
DG(4): I'm not sure . uh .

DG(5): may be you can just press control .
DF(6): I tried that and no luck

The DG does not know that the DF tried the first plan so he suggest it explicitly in (5), to which the DF answers that he already tried.

These examples show how the dialogue structure starts to emerge from the task structure. The dialogue structure is **emergent**. It is impossible to specify in advance what actions each participant is to take in a long conversation. Conversations are created opportunistically piece by piece as the participants negotiate purposes and then fulfill them. Following Clark [1996], we call this the **opportunistic** view of conversation. However, conversations do look planed and goal oriented in retrospect. Viewed as a whole, a conversation consists of a hierarchy of parts: conversation, sections, adjancency pairs, and turns. Where does the structure come from? We argue that the structure of the task in which the conversation is embedded does have a strong impact in the emergent structure of a conversation. What is the status of this structure? In the view of the traditional plan-based models to dialogue (such as BDI, see Chapter 3) it would reflect a plan that the DF and DG agree in order to reach their goals. In the opportunistic view, the structure emerges only as the DF and DG do what they need to do in order to deal with the projects that get proposed during the conversation. The structure is a trace of the opportunities *taken*, not of the opportunities *considered*.

As the conversation above unfolded, it could have taken very different directions depending on what the DPs did. It's easy to see this in the SCARE corpus because all the participants had to perform the same task, however the resulting interactions are quite different. For instance, two other DPs performed the "box moving" task much more efficiently as follows:

DG(1): what we need to do is to move these boxes by pressing .
DG(2): turn around .
DF(3): [turns around]
DG(4): it's gonna be one of these buttons again .
DG(5): and it's the one on the left
DF(6): ok [presses the button]

Interestingly, in conversations the roles can change and the DG himself can take advantage of the clarification potential of his own utterances. The previous dialogue illustrates this point. The DG gives the instruction to "move these boxes" in (1) and knows that the plan to achieve it is to *turn around*, and *look at the buttons*, and *press the left one*. So he uses this implied acts, this lower granularity, to further specify this instruction in (2), (3) and (4).

This emergent structure of the dialogue has been characterized saying that the DG is instructing in a top-down (or pre-order) fashion, first verbalizing a higher action in the hierarchy and then verbalizing the sub-actions [Foster and Matheson, 2008; Foster *et al.*, 2009]. However, in such a view, it's not so easy to explain how roles can switch and, more importantly, why some steps are omitted, that is **tacit**. For instance, in the DG instructions above the sensing

action of looking at the buttons is not made explicit. Also, if the DG had not taken all the initiative in this sub-dialogue the turns could have been taken by the DF as follows:[7]

DG(1): you're gonna wanna move the boxes so you see now there's like two on the left and one on the right
DF(2): so let me guess . like the picture . the buttons move them
DG(3): aha that's true so you wanna turn around so you're facing the buttons
DF(4): [turns around]
DG(5): and you wanna push the button that's on the left
DF(6): ok [presses the button]

In this exchange it is the DF and not the DG the one that makes explicit first the need for pressing a button. In our view of the story, there is not a big difference between the two dialogues above. In fact, we selected the examples so that the parallelism is clear: the utterances can be mapped one by one (though they are not exactly in the same order).[8] Certainly this is not necessary, the utterances can be broken down in many different ways in real conversation. However, the example makes the point that there is a guiding structure behind these conversation, namely the task structure. The task structure opens up opportunities that the DPs can uptake or not. When and how they uptake this opportunities result in the emergent structure of dialogue.

[7]This exchange is actually in the corpus, in a dialogue with a third pair of DPs.

[8]You may have noticed that the utterances in the last dialogue are longer and more articulate; these two DPs were girls, while the rest of the pair of DPs are guys.

Chapter 5

Frolog: A system that maintains *context*

The devil is in the details.

from "Random House Dictionary of Popular Proverbs and Sayings,"
Gregory Titelman

In this chapter I present the system Frolog [Benotti, 2009b] that we will use for our analysis by synthesis of conversational implicatures (in Chapter 6). This system implements a (sort of) text-adventure game. Text-adventures are computer games that simulate a physical environment which can be manipulated by means of natural language instructions issued by the player. The game provides feedback in the form of natural language descriptions of the game world and of the results of the players' commands.

A text-adventure game can be seen as an implementation of an instruction giving task in which the direction giver (DG) is the human player, while the game system implements the direction follower (DF) and the simulated physical environment. As for the simulated physical environment, the system implements the reaction to the execution of an action, signaling to the DF the result of the action if successfully executed or the reason for failure otherwise. As the DF, the system implements the interpretation of instructions (including parsing, reference resolution and public acts identification) and the generation of feedback (including the content determination of the feedback, reference generation and realization). The provided feedback can be positive evidence of grounding (if the action was successful) or negative evidence of grounding (if the action cannot be executed).

In this chapter, we describe the architecture of the simulated physical environment and the DF, which we call Frolog, that are integrated into one system. We present a detailed explanation of all the natural language processing modules that make up Frolog. Chapter 6 concentrates on the module in charge of inferring, in context, the implicatures of the instruction that is being interpreted, namely the bridging module. Finally, Chapter 7 concludes this thesis by discussing the extent to which the framework envisaged in Chapter 4 of this thesis has been implemented in Frolog, as well as future directions this research suggests.

5.1 The text-adventure game Frolog

Text-adventures are computer games that simulate a physical environment. The player manipulates the game world using natural language requests. The game answers describing the

game world and the results of the players' actions. Figure 5.1 shows a possible interaction with Frolog when the game starts.

Player: look
Frolog: a brown couch hold Frolog
Frolog: the couch hold a red apple,
a green apple, a brown frog
Frolog: a drawing-room hold a red table
and the couch
Frolog: the drawing-room has a green wall
Frolog: the drawing-room has a south-exit

Player: look at the apple
Frolog: There is more than one apple.
Frolog: There is the red apple.
Frolog: There is the green apple.

Player: look at the red apple
Frolog: the apple is red and small

Player: eat it
Frolog: Frolog can't do that!
Frolog doesn't hold the red apple

Player: take it
Frolog: Frolog holds the red apple

Player: eat it
Frolog: the red apple is gone

Player: look at the red apple
Frolog: Frolog can't do that!
the red apple is not visible

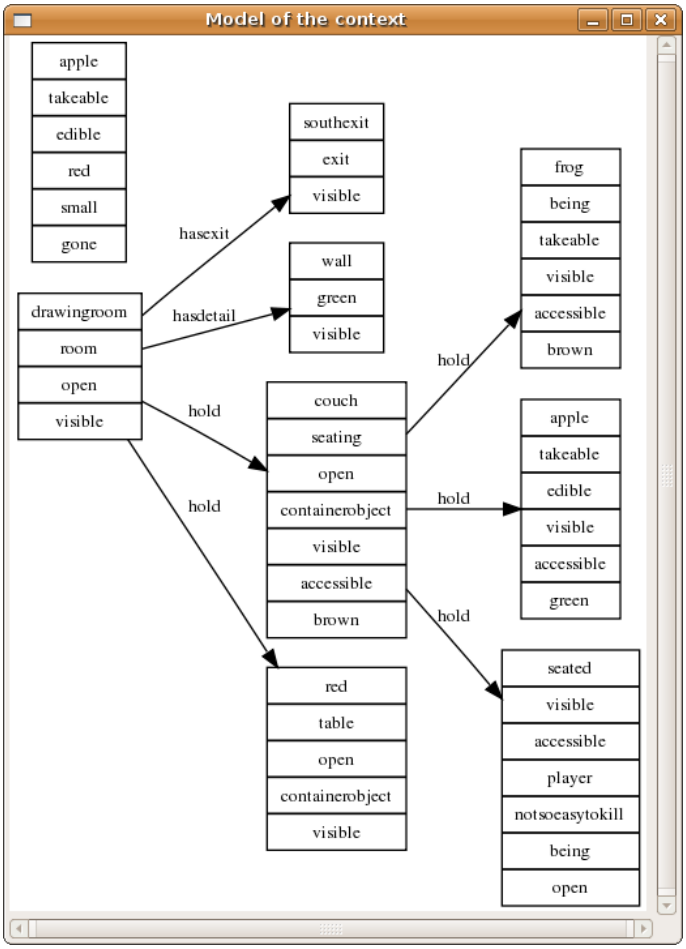


Figure 5.1: A simple interaction with Frolog

The lines in blue were typed by a human player and the lines in black are generated by the system. In the figure, the player starts the interaction by requesting Frolog to perform a **sensing action**, namely *look*. As a result Frolog provides a description generated from the current perspective of the player, which informs the player about the objects that surround him.

The player can also ask Frolog to sense particular objects in the game world and he does so in his second instruction in the figure. However, the instruction is ambiguous so Frolog provides feedback about this ambiguity, mentioning the possible referents of the instructions. In the third instruction the player succeeds in referring to the red apple unambiguously.

In the instruction *eat it* the player refers to the red apple again using a pronoun. However, this instruction involves a **physical action**, namely *eat* which, in this game scenario, requires Frolog to be holding the apple. In general both sensing and physical actions that change the world can have **preconditions** on the state of the game world that must be satisfied in order to execute the action. The player achieves the unsatisfied precondition by requesting the action *take it*. After the apple has been taken the instruction *eat it* can be successfully executed.

The relational model in Figure 5.1 shows Frolog's representation of the interaction when the

conversation ends. The last sensing request by the player cannot be executed, the red apple is no longer visible and this is a precondition of the sensing action *look*. Frolog's interface shows the interaction with the player, the input/output of each module and the model of the context.

Frolog was inspired by a previous text-adventure game called FrOz [Koller *et al.*, 2004]. Frolog still runs on FrOz original game scenarios and simulates the behavior of some of FrOz modules whose performance is not central to the research of this thesis. During this chapter the similarities and differences between Frolog and FrOz will be explained when appropriate.

The rest of the chapter proceeds as follow. The rest of this section explains the general architecture of the system, introducing the processing modules involved and explaining how they interact with the information resources in Frolog. In Section 5.2 we give an introduction to the formalism used by Frolog for knowledge representation and deductive inference; and we also describe the organization of the knowledge bases it uses for representing the game state. Section 5.3 present Frolog's modules in pairs of an NLU module and its NLG counterpart; each pair uses a particular kind of information resource and has analogous input/output. Section 5.4 concludes and links the chapter.

Control and data flow in Frolog

Frolog's architecture is organized in four natural language understanding (NLU) modules: parsing, reference resolution, public acts identification, and internal bridging (i.e. tacit acts identification). There are also four natural language generation (NLG) modules: positive grounding, negative grounding, reference generation and reference resolution. In the version of Frolog that we present in this chapter, the module of bridging is a dummy version that always fails to identify the tacit acts. The addition of genuine bridging capabilities into Frolog is the topic of Chapter 6.

Figure 5.2 shows how the control in the system is passed from one module to the next. The NLU modules parses the command issued by the player (e.g. *Eat the apple*) and constructs a representation of the identified intention of the player. The NLG modules works in the opposite direction, verbalizing the results of the execution of the command (e.g. *The apple is gone*). Frolog uses generic external tools for the most heavy-loaded tasks (depicted in gray in Figures 5.2 and 5.3) namely, a generic parser and a generic realizer for parsing and realization, and an automated theorem prover for knowledge base management; artificial intelligence planners will be used in Chapter 6 for identifying tacit acts. The rest of the modules (depicted in white) were implemented by us in Prolog and Java.

Frolog uses Description Logic (DL) knowledge bases (KB) to codify assertions and definitions of the concepts relevant for a given **game scenario**. Frolog's representation of the **context** is stored in its game scenario. A game scenario is constituted by the elements depicted in light blue in Figure 5.3. We will describe each of the elements in turn.

Frolog uses two knowledge bases, which share the set of **common definitions** of the key concepts in the game world and how they are interrelated. Some of these concepts are basic notions (such as *object*) or properties (such as *red*), directly describing the game world, while

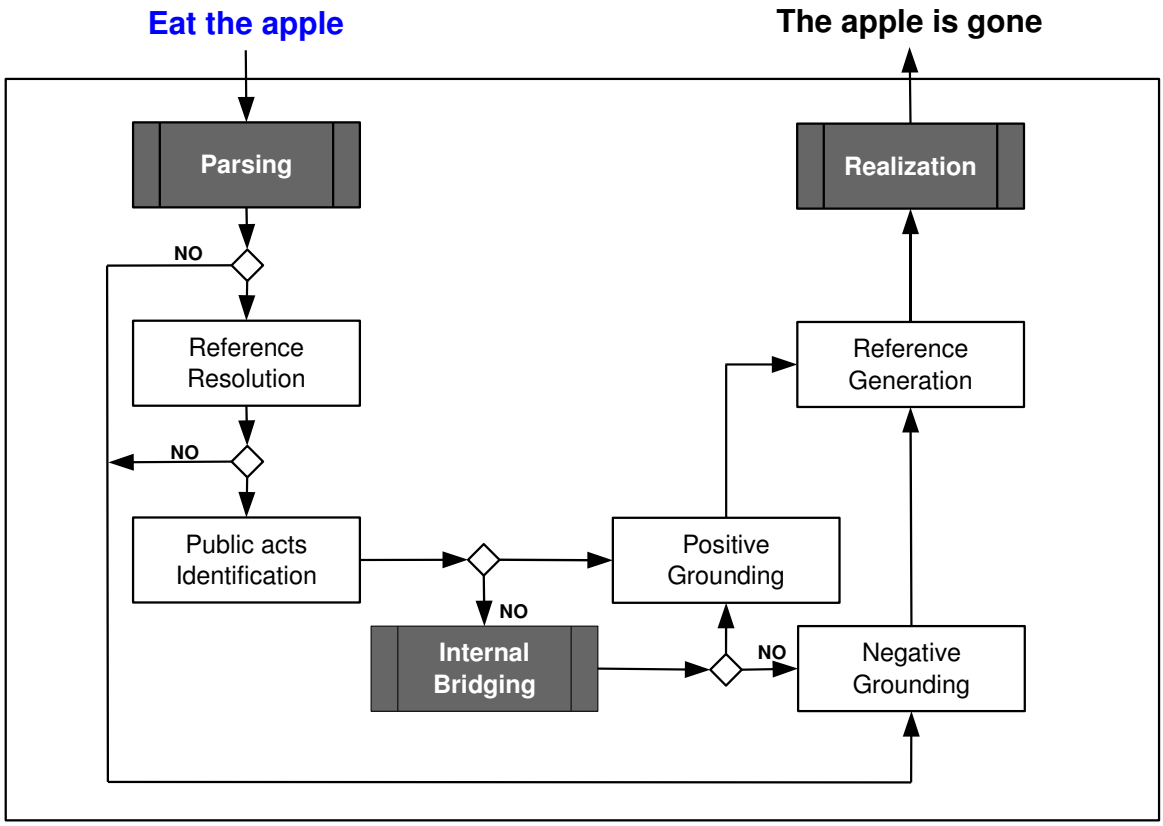


Figure 5.2: Control flow in Frolog

others define more abstract notions like the set of all the individuals Frolog can see (the individuals that are *visible* to Frolog). The knowledge bases specify properties of particular individuals (for example, an individual can be an *apple* or a *player*). Relationships between individuals are also represented here (such as the relationship between an object and its location).

One of the knowledge bases, the **world KB**, represents the *true state* of the game world, while the other, the **interaction KB** keeps track of the Frolog's *beliefs* about the game world (just as the interaction model in Chapter 4 keeps track of the DF beliefs about the world). Again as in Chapter 4, since this interaction KB starts empty, everything that is added here is observed by both Frolog and the player, so we will assume that the information included here is mutually believed between them.

In general, the interaction KB will not contain all the information in the world KB because Frolog will not have explored the world completely, and therefore will not know about all the individuals and their properties.

Most modules make heavy use of deductive inference services in order to query and update the components of a game scenario. Such use is represented as arrows that connect the modules with the KB manager in the Figure 5.3; the direction of the arrow represents whether the module queries or updates the game scenario. The processing modules are independent of particular game scenarios; by plugging in a different game scenario the player can play a different game.

Like FrOz, Frolog uses the theorem prover RACERPRO [Haarslev and Möller, 2001] to query and modify the Description Logic [Baader *et al.*, 2003] knowledge bases according to the instructions encoded in the action database. In spite of its similarities, it is in its practical reasoning

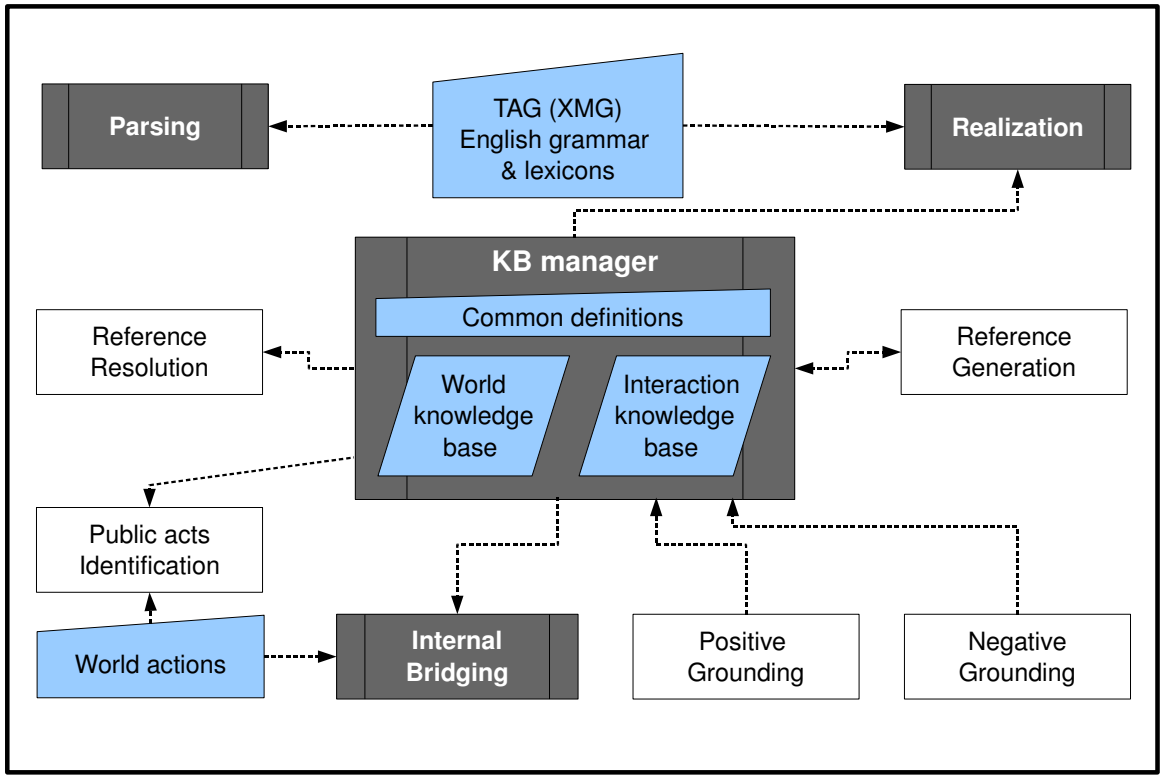


Figure 5.3: Data flow in Frolog

abilities that Frolog differs most from its predecessor FrOz. In Chapter 6 we explore the use of the planners BLACKBOX [Kautz and Selman, 1999] and PKS [Petrick and Bacchus, 2004] for implementing these practical reasoning abilities.

Crucially, a game scenario also includes the definitions of the **world actions** that can be executed by the player (such as the actions *take* or *eat*). Each action is specified (in the action database) as a STRIPS operator (see Chapter 3 for a definition of a STRIPS operator) detailing its arguments, preconditions and effects. The preconditions indicate the conditions that the game scenario must satisfy so that the action can be executed; the effects determine how the action changes the game scenario when it is executed.

Finally, a game scenario also includes the **grammar and lexicons** which are reversibly used for parsing and generation.

5.2 Knowledge representation in Frolog

During this section we briefly introduce the **Description Logic** (DL) [Baader *et al.*, 2003] called $\mathcal{ALCIF}$, the DL used by Frolog to represent the game scenario knowledge bases and to perform automatic inferences over them. Section 5.2.1 includes the basic notions and the formal definitions of the DL $\mathcal{ALCIF}$. Section 5.2.2 explains the deductive reasoning tasks that can be performed on an $\mathcal{ALCIF}$ representation. Finally, Section 5.2.3 describe the content of Frolog's knowledge bases.

5.2.1 The $\mathcal{ALCIF}$ language

The DL research community studies a family of languages for knowledge representation. We will now introduce the syntax and semantics of $\mathcal{ALCIF}$, which is the formalism used by Frolog to codify its knowledge bases. We will also define the deductive inference tasks that are required by most Frolog modules.

$\mathcal{ALCIF}$ syntax

The syntax of $\mathcal{ALCIF}$ is defined in terms of three infinite countable disjoint alphabets. Let CON be a countable set of *atomic concepts*, ROL a countable set of *atomic roles* and IND a set of individuals. Moreover, FUN $\subseteq$ ROL is the set of *functional atomic roles*. We will define the language in three steps. First, we define the $\mathcal{ALCIF}$ operators that let us construct complex concepts and roles from atomic ones.

Definition 10.
An $\mathcal{ALCIF}$ ***role*** *can be:*

- *An atomic role R such that $R \in$ ROL*
- *The inverse of an atomic role: R^{-1} such that $R \in$ ROL*

An $\mathcal{ALCIF}$ ***concept*** *can be:*

- *An atomic concept C such that $C \in$ CON*
- *$\top$, the trivial concept called* top
- *Concepts defined using Boolean operators: Let C and D be two concepts then the following expressions are also concepts: $\neg C$, $C \sqcap D$, $C \sqcup D$*
- *Concepts defined using existential and universal quantified roles: Let C be a concept and R a role then the following are also concepts: $\exists R.C$, $\forall R.C$*

$\mathcal{ALCIF}$ is not very expressive with respect to complex roles, but the language offers a richer operator variety when defining complex concepts. Now, we can specify which are the kinds of definitions that can be included in an $\mathcal{ALCIF}$ knowledge base.

Definition 11. *Given two concepts C and D there are two kinds of* ***definitions****:*

1. Partial Definitions*: $C \sqsubseteq D$. The conditions specified in C are sufficient in order to qualify C elements as D members, but they are not necessary conditions.*
2. Total Definitions*: $C \equiv D$. The conditions specified in C are necessary and sufficient to qualify C elements as D members, and vice-versa. The concepts C and D are equivalent.*

The set of definitions in a knowledge base K is called the **TBox** (*Terminological Box*) and it contains definitions of the primitive and derived notions, and their interrelation. Formally, an $\mathcal{ALCIF}$ TBox is a finite set of $\mathcal{ALCIF}$ definitions. Example 5.1 illustrate this definition presenting a possible fragment of Frolog's TBox.

Finally, we will define the kinds of assertions that can be included in an $\mathcal{ALCIF}$ knowledge base.

DEFINITION 12. ***Assertions*** *let us* assign properties *to particular elements in the domain. Suppose that* $a, b \in \mathsf{IND}$ *are two individuals,* C *is a concept and* R *is a role, there exist two kinds of assertions:*

1. *Assign elements to concepts: the assertion* $a{:}C$ *specifies that the concept* C *is applicable to the element* a*. I.e., all the conditions specified by* C *are applicable to* a*.*
2. *Assign relationships between elements: the assertion* $(a, b){:}R$ *specifies that the elements* a *and* b *are related via the role* R*.*

The set of assertions in a knowledge base K is called the **ABox** (*Assertional Box*) and it contains specific information about certain distinguished individuals in the modeled domain. Formally, an ABox is a finite set of $\mathcal{ALCIF}$ assertions. Example 5.1 illustrate this definition presenting a possible fragment of Frolog's ABox.

$\mathcal{ALCIF}$ semantics

DEFINITION 13. *An* ***interpretation*** *(or model) for the* $\mathcal{ALCIF}$ *syntax is a pair* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$*.* $\Delta^{\mathcal{I}}$ *is the domain, an arbitrary non empty set that can be infinite, while* $\cdot^{\mathcal{I}}$ *is an interpretation function of atomic concepts, atomic roles and individuals such that:*

- *Atomic concepts are interpreted as subsets of the domain: Let* $C \in \mathsf{CON}$ *then* $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$*.*
- *Atomic roles are interpreted as sets of pairs of elements in the domain: Let* $R \in \mathsf{ROL}$ *then* $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$*. Moreover, if* $R \in \mathsf{FUN}$ *then* $R^{\mathcal{I}}$ *is a partial function.*
- *Each individual* $a \in \mathsf{IND}$ *is interpreted as an element in the domain: Let* $a \in \mathsf{IND}$ *then* $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$*.*

Given an interpretation $\mathcal{I}$, the concepts C and D, and a role R we can define the semantic of the three language levels introduced in the previous section. We will begin by extending $\mathcal{I}$ to complex roles and concepts. An arbitrary concept is interpreted recursively as follows:

- $(\top)^{\mathcal{I}} := \Delta^{\mathcal{I}}$
- Boolean operators:

$$(\neg C)^{\mathcal{I}} := \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$$
$$(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$$
$$(C \sqcup D)^{\mathcal{I}} := (\neg(\neg C \sqcap \neg D))^{\mathcal{I}}$$

- Relational operators:[1]

$$(\exists R.C)^{\mathcal{I}} := \{a \mid \exists b.(a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$$
$$(\forall R.C)^{\mathcal{I}} := (\neg(\exists R.\neg C))^{\mathcal{I}}$$

Now we can define when an interpretation:

- $\mathcal{I}$ satisfies a partial definition $C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies a total definition $C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies an assertion $a{:}C$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$

[1] An inverse role is interpreted as follows: $(R^{-1})^{\mathcal{I}} := \{(b, a) \mid (a, b) \in R^{\mathcal{I}}\}$

- $\mathcal{I}$ satisfies an assertion (a, b):R iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

An interpretation $\mathcal{I}$ satisfies a knowledge base $K = \langle T, A \rangle$, such that T is the TBox and A is the ABox (and we write $\mathcal{I} \models K$) iff $\mathcal{I}$ satisfies all the definitions in T and all the assertions in A. K is satisfiable iff there exists an interpretation $\mathcal{I}$ such that $\mathcal{I} \models K$.

EXAMPLE 5.1. The previous definitions are illustrated in the following knowledge base that captures the situation in Figure 5.4

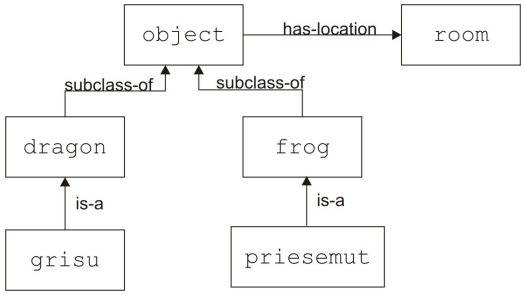


Figure 5.4: A possible fragment of Frolog's game scenario

Let $K = \langle T, A \rangle$ be a knowledge base such that $\mathsf{CON} = \{\texttt{dragon}, \texttt{frog}, \texttt{object}, \texttt{room}\}$, $\mathsf{FUN} = \{\texttt{has-location}\}$, $\mathsf{IND} = \{\texttt{grisu}, \texttt{priesemut}\}$ and:

- $T = \{\texttt{dragon} \sqsubseteq \texttt{object}, \texttt{frog} \sqsubseteq \texttt{object}, \texttt{object} \sqsubseteq \exists \texttt{has-location}.\texttt{room}\}$
- $A = \{\texttt{grisu} : \texttt{dragon}, \texttt{priesemut} : \texttt{frog}\}$

Under the formal semantics we have just introduced we can verify that K is satisfiable, i.e. it has at least one model $\mathcal{I} = \langle \{\texttt{grisu,priesemut,x,y}\}, \cdot^{\mathcal{I}} \rangle$ such that:

$(\texttt{dragon})^{\mathcal{I}} = \{\texttt{grisu}\}$,
$(\texttt{frog})^{\mathcal{I}} = \{\texttt{priesemut}\}$,
$(\texttt{object})^{\mathcal{I}} = \{\texttt{grisu,priesemut}\}$,
$(\texttt{grisu})^{\mathcal{I}} = \texttt{grisu}$,
$(\texttt{priesemut})^{\mathcal{I}} = \texttt{priesemut}$, and
$(\texttt{has-location})^{\mathcal{I}} = \{(\texttt{grisu,x}), (\texttt{priesemut,y})\}$.

5.2.2 Deductive reasoning in Frolog

The notion of satisfiability of a knowledge base that we defined in the previous section is one of the basic reasoning tasks in DL. Given that the knowledge base codifies the information that we know about certain domain, it is at least required that this information is consistent, i.e. satisfiable by at least one model. But apart from checking whether the knowledge base is consistent, we need methods that let us query the information that is implicit in the knowledge base.

We can now define the following standard inference tasks. Let K be a knowledge base, C and D two concepts, R a role and $a, b \in \mathsf{IND}$, we can define the following inference tasks with respect to a knowledge base:

- **Subsumption**, $K \models C \sqsubseteq D$. Verifies whether for all interpretation $\mathcal{I}$ such that $\mathcal{I} \models K$ we have that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.
- **Instance checking**, $K \models a : C$. Verifies whether for all interpretations $\mathcal{I}$ such that $\mathcal{I} \models K$ we have that $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- **Relation checking**, $K \models (a, b) : R$. Verifies whether for all interpretation $\mathcal{I}$ such that $\mathcal{I} \models K$ we have that $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.
- **Concept Consistency**, $K \not\models C = \neg\top$. Verifies whether for some interpretation $\mathcal{I}$ such that $\mathcal{I} \models K$ we have that $C^{\mathcal{I}} \neq \emptyset$.

EXAMPLE 5.2. Given the knowledge base K defined in the Example 5.1, it is possible to infer further information. For instance, we can infer that the concept `object` is consistent with respect to K: there exists some interpretation that satisfies K and that assigns a non empty interpretation for `object`.

The basic reasoning tasks can be used to define more complex ones that are useful for implementing applications, such as:

- **Retrieval:** for a given concept C, find all the individuals mentioned in the knowledge base that are instances of C.
- **Most specific concepts:** for a given individual a mentioned in the knowledge base, find the most specific concepts in the knowledge base such that a is a member of them.
- **Immediate ancestor (descendant) concepts:** for a given concept C, find all the concept immediately above (under) C in the hierarchy defined by the knowledge base.

For the DL $\mathcal{ALCIF}$, the inference tasks we defined in this section are very complex. For example, subsumption checking of two concepts with respect to an arbitrary knowledge base is a complete problem for the complexity class EXPTIME (exponential time).

RACERPRO is the DL reasoner that is used inside Frolog. RACERPRO was developed at the University of Hamburg by Haarslev and Müller. It is implemented in COMMON LISP and it is available for research purposes as a server program that can be used both in Windows and Linux. RACERPRO offer inference services to client applications through an http interface. Recently, RACERPRO has became a commercial product that sells its services through the RACER SYSTEMS GmbH & Co company. Moreover, there are implementations of graphical interfaces for taxonomy editing that can connect with RACERPRO.

RACERPRO offers different inference services including the ones described in Section 5.2.2. Moreover, this inference engine supports multiple TBoxes and ABoxes. Furthermore, it is one of the few systems that allows for the addition and retraction of ABox assertions even after queries have been performed. All these characteristics are crucial for Frolog performance and motivate the choice of RACERPRO as inference service provider.

5.2.3 Frolog's knowledge bases

As we mentioned before, most of the information defining a particular Frolog scenario is encoded as DL knowledge bases. In fact, underlying the system there are two knowledge bases, which share a set of common definitions represented in the TBox; and differ only in their set of assertions, that is in the ABoxes. The common TBox defines the key notions in the world and how they are interrelated. The **world ABox** represents the *true state* of the world, while the **interaction ABox** keeps track of the Frolog's *beliefs* about the world.

The ABoxes specify the kind of an individual (for example, an individual can be an *apple* or a *player*) detailing to which concept an individual belongs. Relationships between individuals in the world are also represented here (such as the relationship between an object and its location). A fragment of an example ABox describing a possible state of the world in the Fairy Tale Castle scenario is:

`room(empfang)`	`alive(myself)`
`player(myself)`	`alive(worm1)`
`frog(priesemut)`	`alive(priesemut)`
`crown(crown2)`	`has-location(myself,couch1)`
`apple(apple1)`	`has-location(priesemut,table1)`
`apple(apple2)`	`has-location(apple1,couch1)`
`worm(worm1)`	`has-location(apple2,couch1)`
`couch(couch1)`	`has-location(couch1,empfang)`
`table(table1)`	`has-location(table1,empfang)`
`exit(drawing2treasure)`	`part-of(crown2,priesemut)`
`has-exit(empfang,drawing2treasure)`	`part-of(worm1,apple1)`
`green(apple1)`	

A graphical representation of the relations represented in this ABox fragment is given in Figure 5.5. Individuals are connected to their locations via the `has-location` role. Objects are connected with things they are part of via the role `part-of` (e.g., `part-of(worm1,apple1)`).

The TBox specifies that the world is partitioned into three main concepts: generic containers, objects and things that can be opened and closed. Properties that can change in the world such as `alive` are also defined as concepts. The TBox contains as well axioms that establish a taxonomy between concepts such as:

`takeable` $\sqsubseteq$ `object`
`apple` $\sqsubseteq$ `takeable`
`exit` $\sqsubseteq$ `open-closed`
`room` $\sqsubseteq$ `generic-container`
`player` $\sqsubseteq$ `generic-container`

A graphical representation of a TBox fragment for Frolog Fairy Tale Castle scenario is given in Figure 5.6.

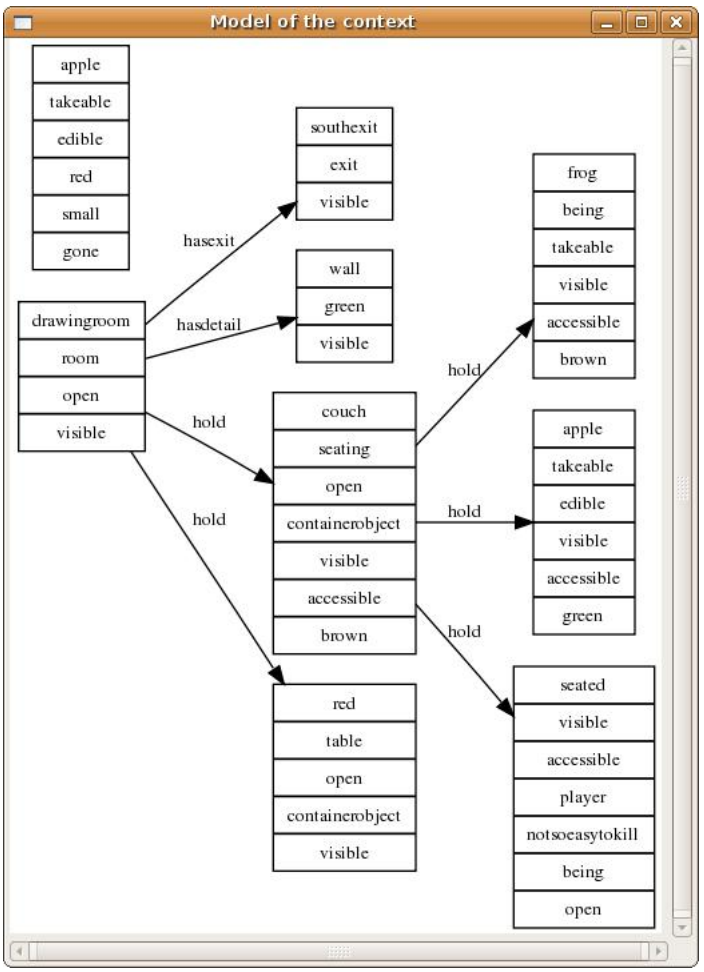


Figure 5.5: Graphical representation for roles in Frolog's ABox

On top of the basic definitions, the TBox specifies more abstract concepts that are useful in the game context. For example, the concept `here`, which contains the room in which the player is currently located, is defined as:

$$\texttt{here} \equiv \exists \texttt{has-location}^{-1}.\texttt{player} \tag{5.1}$$

In the example ABox we introduced for the Fairy Tale Castle scenario, `here` denotes the singleton set `couch1`. It is the only individual to which an instance of `player` is related to via the role `has-location`. Another important concept in the game is `accessible`, which contains all individuals that Frolog can manipulate.

$$\begin{aligned}
\texttt{accessible} \equiv \quad & \texttt{here} \sqcup && (5.2)\\
& \exists \texttt{has-location.here} \sqcup && (5.3)\\
& \exists \texttt{has-location.(accessible} \sqcap \texttt{open)} \sqcup && (5.4)\\
& \exists \texttt{part-of.accessible} && (5.5)
\end{aligned}$$

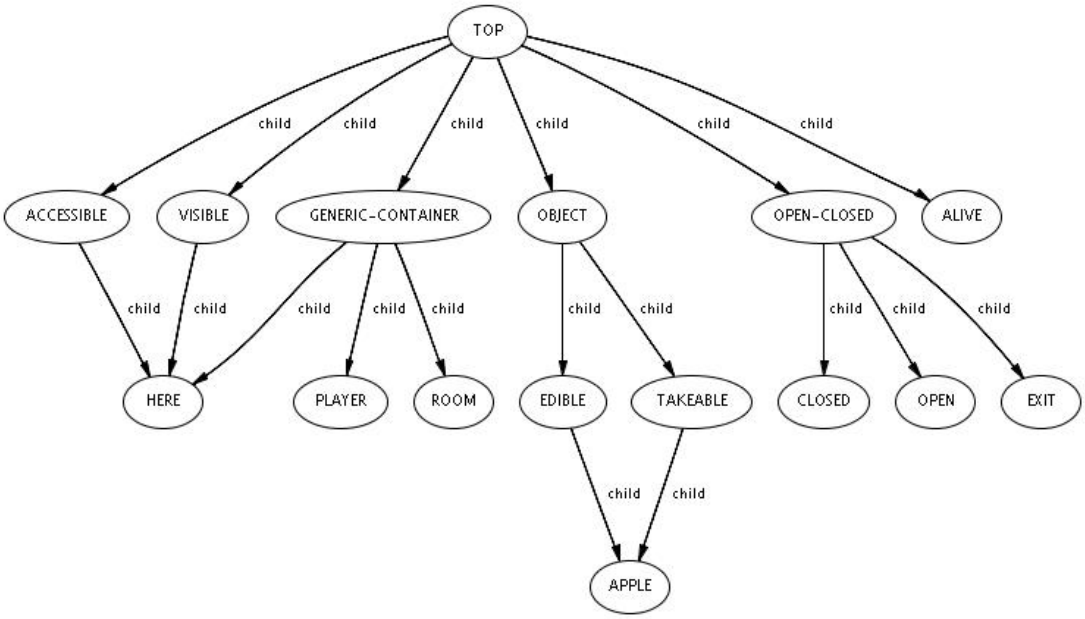


Figure 5.6: Graphical representation of a fragment of the game TBox

This DL definition means that the location where Frolog is currently standing is accessible to him, as well as the individuals that are in the same location. If such individual is some kind of container and it is open then its contents are also accessible; and if it has parts, its parts are accessible as well. In the Fairy Tale Castle ABox we introduced, the denotation of `accessible` will include the set {`couch1,myself,apple1,apple2,worm1`}.

Finally, the concept `visible` can be defined in a similar way as `accessible`. The definition is a bit more complex, incorporating more individuals and is intended to denote all individuals that Frolog can see from his position in the game world. In the Fairy Tale Castle ABox we introduced, `visible` denotes the set of all the objects in the world ABox.

5.3 Frolog's knowledge processing modules

This section presents Frolog's modules in pairs of an NLU module and its NLG counterpart; each pair uses a particular kind of information resource and has analogous input/output.

5.3.1 Reversible parsing and realization

The parsing and the realization use modules the same linguistic resources, namely a **reversible meta-grammar**, a lemma lexicon, and a morphological lexicon represented in the **XMG grammatical formalism** [Crabbé and Duchier, 2005]. So parsing and realization in Frolog are fully reversible, that is, Frolog can parse everything that it generates and vice-versa.

The meta-grammar used specifies a **Tree Adjoining Grammar** (TAG) of around 500 trees and integrates a semantic dimension à la [Gardent, 2008]. An example of the semantics

associated with the player input "open the chest" is depicted in Figure 5.7. This semantic representation is the format of the output of the parsing module and the input of the realization module.

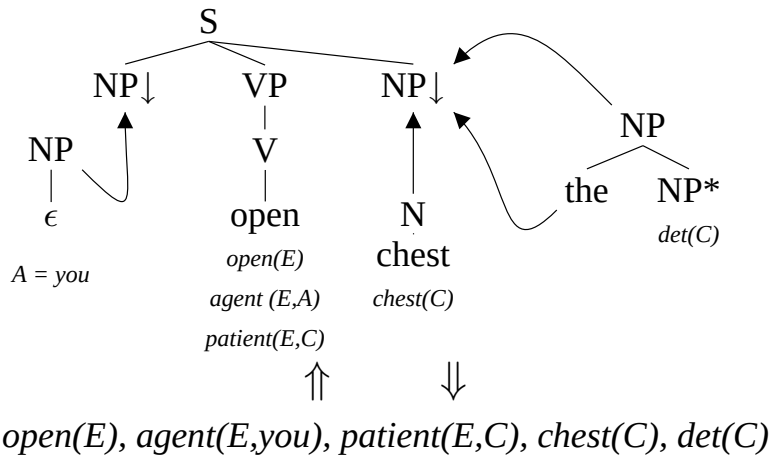


Figure 5.7: Parsing/realization for "open the chest"

The **parsing module** performs the syntactic analysis of a command issued by the player, and constructs its semantic representation using the TAG parser TULIPA [Kallmeyer *et al.*, 2008] (illustrated in the Figure 5.7 by ⇓). The **realization module** works in the opposite direction, verbalizing the results of the execution of the command from the semantic representation using the TAG surface realizer GENI [Gardent and Kow, 2007] (illustrated in the Figure 5.7 by ⇑).

The interfaces to the parser TULIPA and the generator GENI were implemented in Java; the linguistic resources are loaded when the game starts and then the parser and generator are used as servers when an utterance has to be parsed or some content has to be verbalized.

Although GENI and TULIPA use different formats for the grammatical resources they expect, Frolog automatically generates the formats expected by them from a single source. To the best of our knowledge, Frolog is the first implemented system which actually uses a general purpose parser and a general purpose generator in a fully reversible fashion (using the architecture proposed in [Kow *et al.*, 2006]). The general architecture of the required format conversions (illustrated in Figure 5.8) that was implemented should be readily integrable into other systems that require English or French parsing and realization.

The files depicted in white in Figure 5.8 (the Source XMG grammar and the source lemma lexicon) are the ones that include the source information about the grammar and lexicon. The files depicted in gray are automatically generated. Frolog uses the tools METATAG and LEXCONVERTER [Parmentier, 2007] to convert between formats.

Frolog uses the XMG grammar for English called XMG-based XTAG that is described in [Alahverdzhieva, 2008].[2] Its design is based on the **XTAG grammar**, a project to develop a wide coverage grammar for English at UPENN [Group, 2001]. However, Frolog's grammar is

[2]Documentation about the grammar can be found in http://svn.devjavu.com/katya/XMG-basedXTAG/.

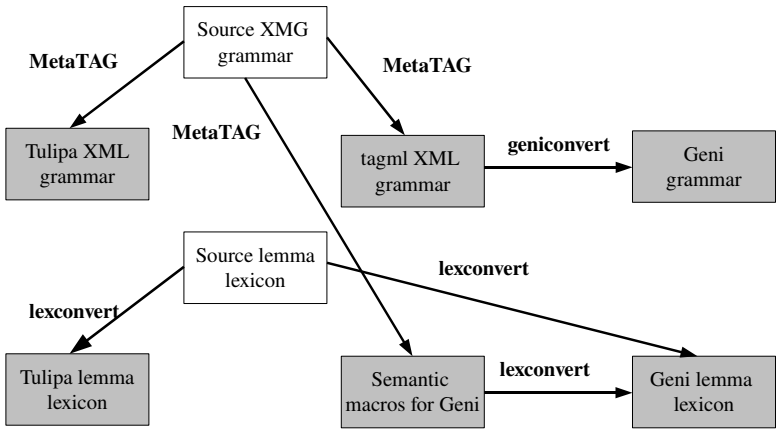


Figure 5.8: The general reversible architecture for Tree Adjoining Grammars

not a TAG grammar but a meta-grammar that is factorized in several ways using the formalism XMG for specifying linguistically motivated classes. For instance, the tree in Figure 5.7 integrates the TAG tree in Figure 5.9 obtained from the XMG grammar and anchored with the lexeme "open".

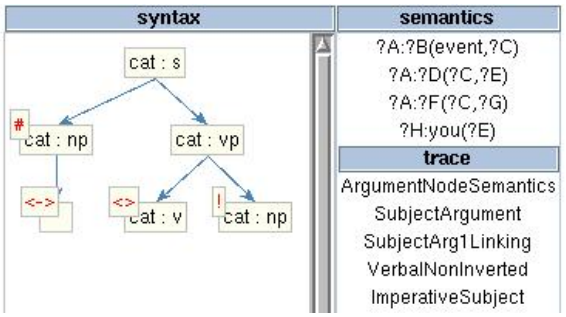


Figure 5.9: TAG tree for transitive verbs (MetaTAG graphical user interface)

The TAG tree in Figure 5.9 is specified in several different XMG classes which are listed in the trace field of the screen-shot. The classes defined in the XMG grammar and the kind of factorization used by the designer of the grammar turns out to be crucial for using the generator GenI in a deterministic way.

Once that GenI starts, clients can connect to it through sockets. So, from what we've seen so far in order to generate "the red apple is big" you should send the following semantics to GenI.

```
[def(A) red(A) apple(A) big(A)]
```

However, when we send this semantics to GenI and ask it to produce a sentence, it produces two outputs:

```
[the red apple is big, the apple is red and big]
```

This makes sense given the grammar: there are two ways of realizing this semantics. It is possible to instruct GenI to produce only one of these realizations. The way this is done is by enriching the input semantics with the name of the XMG classes that we want GenI to use for the realization. Since the XMG classes allow for factorization and inheritance the class specified here can be generic. And then it offers a way to implement, for instance, the **topic-focus** distinction [Gundel and Fretheim, 2004]. We do this in Frolog in a naive way but, we believe, well founded way. Our approach consists in indicating that those syntactic classes that are usually part of the topic (e.g. classes that constitute the subject) inherit from a generic XMG class called `topic`; and those syntactic classes that are usually par of the focus (e.g. classes that constitute the object) inherit from a generic XMG class called `focus`. Having said this then the way to obtain only "the red apple is big" is by enriching the semantics above as follows:

```
[def(A) red(A)[topic] apple(A)[topic] big(A)[focus]]
```

That is, you just associate with the semantic literal the name of the XMG class that you want GenI to use. This method for deterministic realization is already theoretically explored in [Gardent and Kow, 2007]. However, Frolog offers the ideal setup to test it because of its internal representation of context. That is, if a literal marked as `topic` in the semantics to be realized is not contained in the *interactional knowledge base*, then Frolog is aware that he is presupposing something he didn't say before. Similarly, for focus, if a literal marked as `focus` in the semantics to be realized is in fact already contained in the *interactional knowledge base* then Frolog is aware that he is asserting something he already said. Such knowledge can be used to experiment with different realization strategies that play with the topic-focus distinction and its relation to the interactional context. We think this is a promising line of work which we add to our list of future work.

5.3.2 Reference resolution and generation

The **reference resolution** (RR) module is responsible for mapping the semantic representations of definite and indefinite noun phrases and pronouns to individuals in the knowledge bases (illustrated in Figure 5.10 by ⇓). The reference generation (RG) module performs the inverse task, that is it generates the semantic representation of a noun phrase that uniquely identifies an individual in the knowledge bases (illustrated in the Figure 5.10 by ⇑).

Frolog uses the theorem prover RacerPro [Haarslev and Möller, 2001] to query the KBs and perform RR and RG. In order to manage the ambiguity of referring expressions two levels of saliency are considered. The interaction KB is queried (instead of the world KB) naturally capturing the fact that the player will not refer to individuals he doesn't know about (even if they exist in the world KB). Among the objects that the player already knows, a second level of saliency is modeled employing a simple stack of discourse referents which keeps track of the individuals most recently referred to. A new individual gets into the interaction KB when the player explores the world.

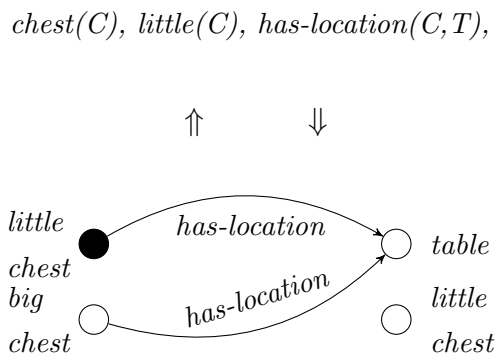


Figure 5.10: RR/RG for "the little chest on the table"

The algorithms used for RR and RG are similar to those in [Koller *et al.*, 2004] and are described in the following subsections.

Reference resolution

In our example, the resolution of a *definite* description is needed (the apple); this amounts to finding a unique entity which, according to the player knowledge, is visible and matches the description. To compute such an entity, a DL concept expression corresponding to the description is constructed and then a query is send to RACERPRO asking for all instances of this concept. In the case of our example, all instances of the concept

$$\texttt{apple} \sqcap \texttt{visible} \sqcap \texttt{green} \tag{5.6}$$

would be retrieved from the player knowledge base. If such a query yields only one entity ({`apple1`} for "the green apple" from the knowledge base in the example ABox we introduced in the previous section), the reference has been unambiguous and succeeds. It may be the case, however, that more than one entity is returned. For instance, (given the same knowledge base) the query for 'the apple' would return the set {`apple1, apple2`}.

In such a situation, referents which are not *salient* according to the actual discourse model are filtered out. If this narrows the candidate set to a singleton, we are done. Otherwise, we assume that the definite description was not unique and return an error message to the player indicating the ambiguity.

To resolve *indefinite* NPs, such as 'an apple', the player knowledge base is queried as described above. However, unlike the definite case, a unique referent is not required in this case. Instead it is assumed that the player did not have any particular object in mind and one of the possible referents is arbitrarily chosen (and the choice will be informed to the player by the generation module). After all references has been resolved, a ground term (or sequence of ground terms) that specifies the action (or actions) intended by the player is passed to the Actions module. In our example, this representation would be: `[[take(frolog,apple1)]]`.

Since reference resolution is an isolated module, which does not receive feedback from the

action identification modules then the interpretations that it does not take into account the **affordabilities** [Steedman, 2002] of the situation. Therefore, situations such as the ones illustrated in Figure 5.11 happen in FROLOG. The use of definites is already problematic but the problem becomes even more evident with indefinites.

Player: go through the north exit
Frolog: [Frolog go through the north exit into the drawing room]
Frolog: the drawing room hold you
Frolog: the north exit leads to the drawing-room
Frolog: the drawing room holds Frolog, the table and the couch
Frolog: the drawing room has the red wall
Frolog: the drawing room has the south-exit

Player: look at the couch
Frolog: the couch is brown
Frolog: the couch holds the red apple
Frolog: the couch has a leg, a leg, a leg and a leg

Player: take the red thing
Frolog: There is more than one red object.
Frolog: There is the red apple.
Frolog: There is the red wall.

Player: take the red wall
Frolog: Frolog can't do that!
The red wall is not takeable

Player: take a red thing
Frolog: Frolog can't do that!
The red wall is not takeable

Player: go to the treasury
Frolog: You can't see any treasury

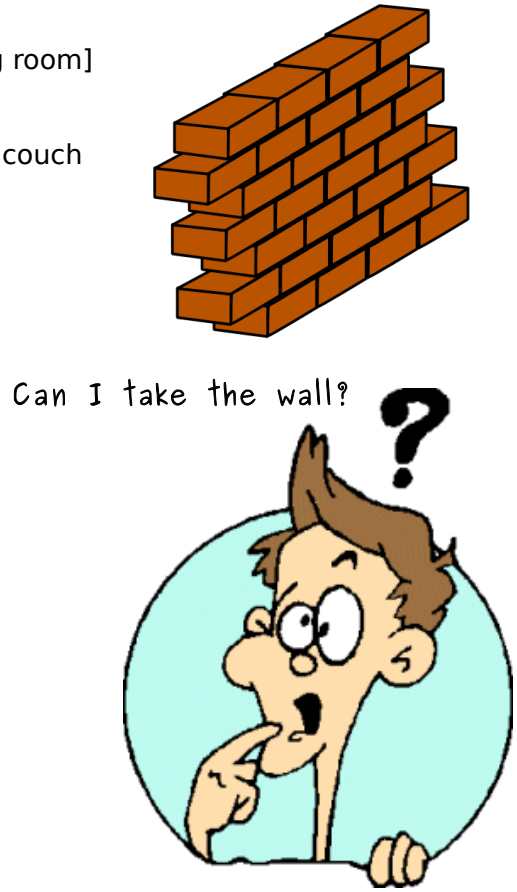


Figure 5.11: affordability and referring

Reference generation

To refer to an object that the player already has encountered, a definite description is constructed that, given the player beliefs, uniquely identifies this object. The properties of the target referent are looked in some predefined order (e.g., first its type, then its color, its location, parts it may have, and so on). A property is added to the description if at least one other object (a *distractor*) is excluded from it because it doesn't share this property. This is done until the description uniquely identifies the target referent. Once more, RACERPRO queries on the player ABox are used to compute the properties of the target referent and the distracting instances, and to check whether a given property is of a certain kind (e.g., color).

Following the cycle of our original example, "Take the green apple", the content that needs to be verbalized for it is:

```
[apple(apple1) def(apple1) green(apple1)
 has-location(apple1 myself)]
```

The reference generation task is simpler for objects which are new to the player (newness can be determined by querying whether the individual is mentioned in the player ABox). In this case, an indefinite NP containing the type and (if it has one) color of the object is generated. RACERPRO retrieval functionality is used to extract this information from the world ABox.

In our example "Look at the green apple", if the player knows that there are two apples and that the second apple is red but she doesn't know about the worm, the second sentence to verbalize would be enriched as follows:

```
[has-detail(apple1 worm1) worm(worm1) apple(apple1)
 undef(worm1) def(apple1) green(apple1)])
```

The message now contains the information that an indefinite reference to `worm1` should be built, referring to it as "a worm". `apple1` should be referred to by the definite description "the green apple". The color was added to distinguish it from the other apple which is red.

5.3.3 Public act identification and grounding

Here we briefly explain the basic behavior of three of the last four modules in charge of act identification and feedback, namely, public act identification, and positive and negative grounding feedback. The remaining module, namely **internal bridging** is the main focus of this thesis and the topic of Chapter 6. In Chapter 6 we explain how internal bridging is implemented and how its integration into Frolog impacts on the grounding modules.

These four last modules share the last information resource that constitute an scenario, namely, the action database. The action database includes the definitions of the actions that can be executed by the player (such as *take* or *open*). Each action is specified as a STRIPS-like operator [Fikes *et al.*, 1972] detailing its arguments, preconditions and effects as illustrated below.

Public act identification

Here we are going to explain in detail how an action made public by the player is interpreted. To illustrate our explanation, let us consider a concrete input and analyze how it is handled by the system. Suppose that the player has just said "Take the key." The semantic representation of this command (obtained by the language understanding module) will be the ground term `take(key1)` (where `key1` represents the only key that the player can see in the current state of the game). This ground term will be passed to the next processing module in the architecture.

When a ground term is received by the action handling module, it is matched against the list of action schemes. The action schema that will match the ground term of our example is:

```
action:
  take(X)
preconditions:
  accessible(X),
  takeable(X),
  not(in-inventory(X))
effects:
  add: in-inventory(X)
  del: has-loc(X indiv-filler(X has-loc))
player effects:
  add: in-inventory(X)
  del: has-loc(X indiv-filler(X has-loc))
```

The term `X` in the above schema is a variable that gets bound to the actual argument of the action. In our example, `X` would be bound to the constant `key1`, and thus the preconditions and effects will become ground terms. Once the action schema is instantiated, it is time to check that the action can be executed. An action can be executed if all its preconditions are satisfied in the current world KB. The preconditions can require that individuals belong to certain concepts or that they are related by certain roles. For example, the execution of the action `take(key1)` requires that the key is accessible to the player (`accessible(key1)`), that it is small enough to be taken (`takeable(key1)`) and that it is not carried by the player already (`not(in-inventory(key1))`). The theorem prover RACERPRO is used to query the current *world KB*, thereby checking that the preconditions are satisfied.

If the action can be executed, the *world KB* is updated according to the effects of the action. In our example, the key will no longer be in its original location but it will be carried by the player. The original location of the key is obtained by sending the query `indiv-filler(key1 has-loc)` to RACERPRO. A RACERPRO query is embedded in an action schema when the action depends on properties of individuals not explicitly mentioned in the player command (such as the location of the key).

Once the size of the task actions is fixed, as happened when we defined the actions in Frolog's action database, the line between explicatures and implicatures is fixed in Frolog. If an instruction uttered by the player is not a complete action according to the definition of the actions in the world actions database, Frolog tries to infer the missing material. If the material is inferable then we say that such material are **explicatures** of the player instruction. For example, in Figure 5.12 the `source` of `take` is implicit and inferable, because the frog is located in one place known by the Frolog. The `goal` of `put` is required but non-inferable, because there are many places where the frog can be put. Frolog can produce negative grounding such as *Frolog doesn't know where* when an action is ambiguously underspecified.

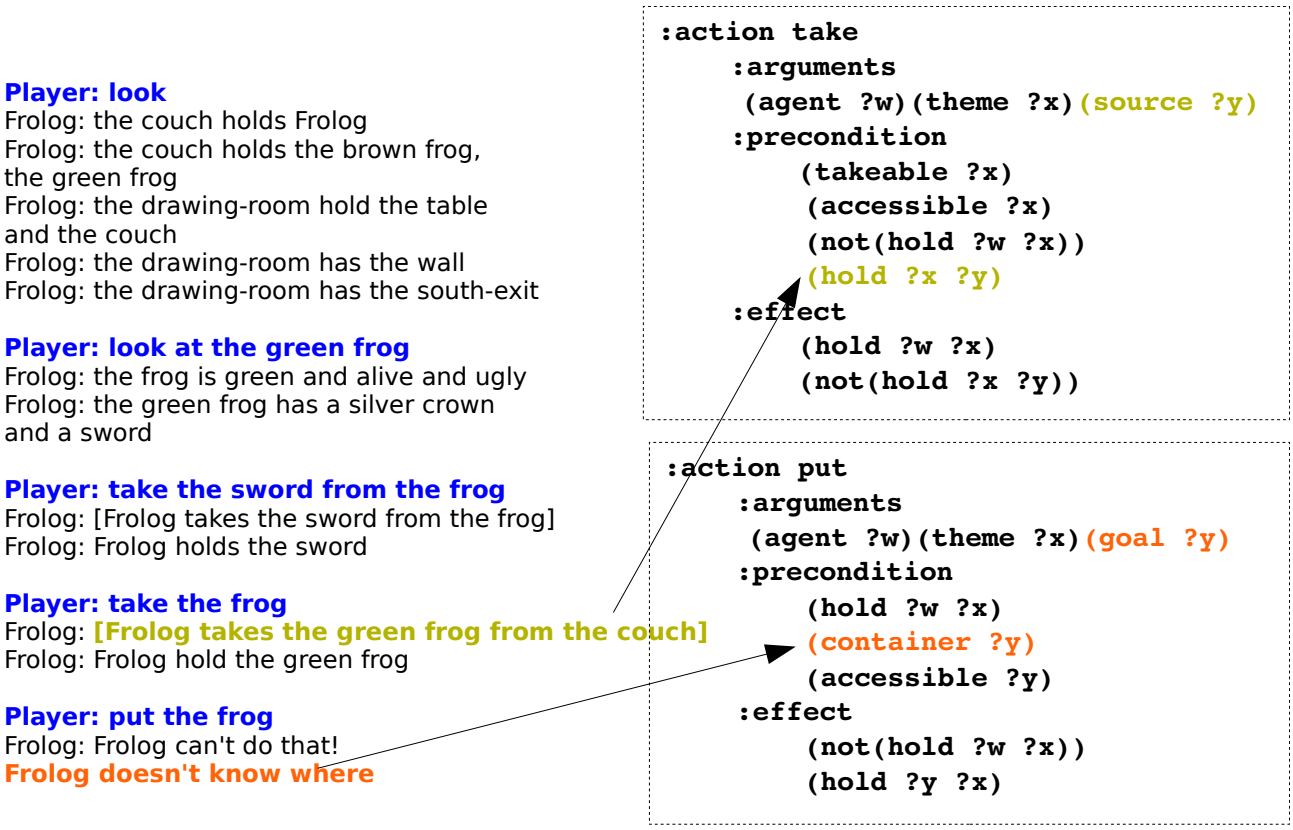


Figure 5.12: Inferable and non-inferable implicit arguments

Positive and negative grounding

The term **grounding** has been traditionally used [Clark, 1996] to describe the process by which interlocutors establish some information as part of the **common ground** (also known as *mutual knowledge*). This definition of grounding is broad enough to allow for grounding to occur at all the levels of communication of the communicative **action ladder** (see Chapter 2 for a definition of action ladder).

In Frolog we call **positive grounding** to the process by which the hearer gives evidence of *successful* establishment of common ground. And we call **negative grounding** to the process by which the hearer gives evidence of *obstacles that prevent* the establishment of common ground.

Once Frolog executed the action, the player needs to know that the action succeeded. To this end, the positive grounding module communicates the effects produced by the action to the player and asserts the effects in the *interaction KB*. If the command cannot be executed in the current game scenario, the first precondition that failed is communicated to the player by the negative grounding module and both KBs remain unchanged. Both situations are illustrated by the Figure 5.13.

Now that we know how the actions module handles a simple action, let us explain how *ambiguous commands* and *tacit actions* are handled in Frolog.

The input of the action module is not a single ground term but a list of possible readings of the input sentence. The list will contain exactly one reading only if the sentence is not

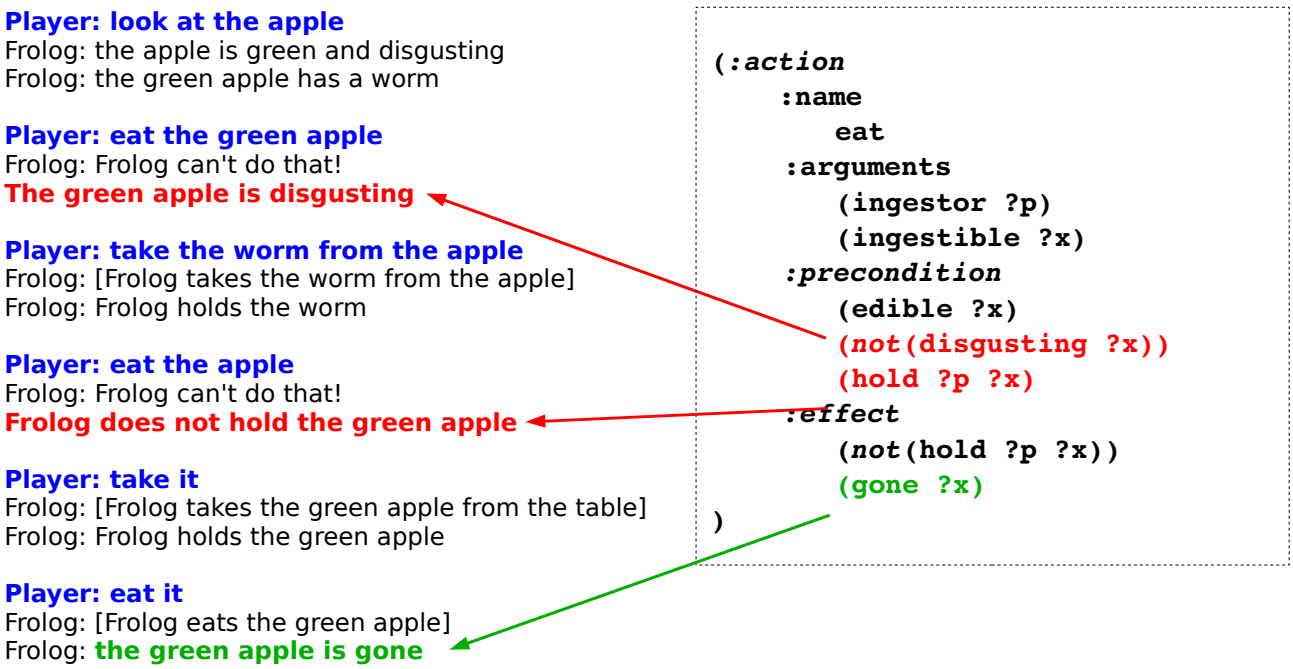


Figure 5.13: Positive and Negative Grounding Evidence Frolog

ambiguous (as in the example in the previous section). Otherwise, the list will contain one entry for each different reading. For example, the sentence "Unlock the door with the key" is syntactically ambiguous and has two possible readings, one in which the propositional phrase "with the key" modifies the verb "unlock" and another in which it modifies the noun phrase "the door." Sentences can also be referentially ambiguous. For instance, the sentence "Take it" has as many readings as there are salient referents in the game scenario. Each reading is itself a list which represents a sequence of actions to be performed one after the other. For example, every reading of the sentence "Take the key and unlock the door with it" will contain two ground terms, one for each action in the sequence.

If the input sentence has more than one reading, Frolog decides among them by trying each action sequence in parallel. When an action fails, the entire reading it belongs to is discarded. For example, the reading of the command "Take it and eat it" which resolves both occurrences of "it" to a key, will be discarded because a key is not edible, although it can be taken.

If only one reading succeeds, the game assumes that this is the command the player had in mind, and commits to the end result of the sequence. If more than one sequence is possible, the game reports an unresolved ambiguity. For instance, the game will report an ambiguity if both readings of the command "Unlock the door with the key" are executable.

5.4 Concluding and linking the chapter

This chapter introduces Frolog, an interactive system that represents the interaction context that is built between a human user and the agent Frolog situated inside an adventure game. The Section 5.1 starts by introducing the general behavior of the system to then get into the innards of it: the control flow and data flow inside the system. In Section 5.2 we give

an introduction to the formalism used by Frolog for knowledge representation and deductive inference; and we also describe the organization of the knowledge bases it uses for representing the context of the interaction. Section 5.3 presents Frolog's modules in groups of modules which share a particular kind of information resource and has analogous input/output. The use of the interaction knowledge base (that is, Frolog's representation of context) turns out to be crucial for almost all the processing modules (even for realization).

Chapter 6

Implicating *interactively* with Frolog

This chapter is organized in four sections. Section 6.1 introduces the chapter linking Frolog's bridging abilities back to the concepts of granularity (Chapter 1), clarifications (Chapter 2), practical inference (Chapter 3) and interactive implicature (Chapter 4). Section 6.2 presents the implementation of Frolog's bridging abilities using classical planning and Section 6.3 explores extensions to this implementation that use planning with knowledge and sensing. Section 6.4 concludes the chapter.

6.1 No implicatures without practical inference

Let's start right away with a sample interaction of the system we presented in Chapter 5. The interaction with the system is reproduced in the Figure 6.1. Let's walk through the example.

Frolog is in a room with a locked door, sitting on a big couch. The player is asking Frolog to look around in turns (1) to (6) reproduced there. She is trying to find a way to unlock the door, when Frolog says that there is a golden key on a table in the room, in turn (6). Then, in turn (7), the player inputs the command "Unlock the door with the golden key." This command cannot be directly executed in the game world because one of the preconditions of the action "unlock" does not hold in the game world, namely `hold(frolog key1)` (as you can verify in the specification of the action unlock on the right). This failed precondition is made explicit by Frolog in turn (9). When the player tries to achieve this failed precondition, by explicitly requesting Frolog to take the key in turn (10), another precondition comes in the way. This time, the command cannot be directly executed because the precondition `(accessible key1)` does not hold in the world (as Frolog announces in turn (12)). The key is not accessible because it is on the table and Frolog is sitting on the couch. Again, the player is forced to explicitly ask Frolog to achieve this precondition and he does so in turn (13) with the command "Stand up". This command can be directly executed on the game world because its preconditions are satisfied (namely, Frolog is seated). As a result, the game world is changed with the effects of the action, which are made explicit in turn (15). Once Frolog is standing in the drawing room, the key is accessible (according to how the accessibility relation is defined in Frolog, see

Chapter 5) and the command "take the key" in turn (16) can be directly executed. Once again the game world is changed with the effects of this action which are made explicit by Frolog in turn (18). In this new state of the game world, all the preconditions of the action "unlock the chest with the golden" hold and this command can be directly executed, as turns (19) to (21) show.

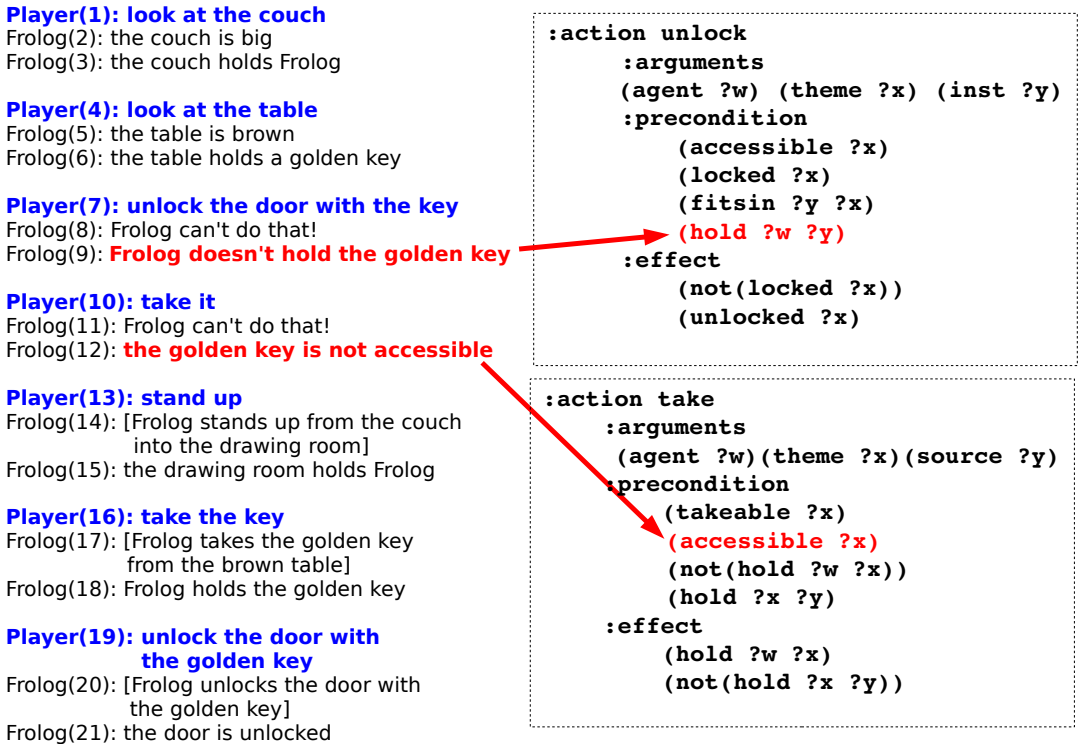


Figure 6.1: Frolog *without* bridging abilities

This is a pretty long interaction in order to get a door unlocked. What's going on in this example? Following the intuitions of Chapter 1 we see that *Frolog cannot switch among granularity levels*. That is, Frolog cannot reconstruct on its own the fine-grained level of granularity intended by the player's commands. Also, using the definitions from Chapter 2 we can observe that *Frolog is making CRs in level 4* in the turns (9) and (12).[1] These CRs are forcing the player to be the one to switch to the lower level of granularity required by the task, that is, they are forcing the player to make explicit the implicated premises of the command. Therefore, what is going on in this example, in the interactive view of implication that we argued for in Chapter 4, is that *Frolog is forcing all the bridging to be externally done by the player*. Frolog is not doing any bridging itself.

From Section 4.2.1, we know that there are at least three reasons why bridging of implicated premises may become external: (a) the inferred implicated premises turn out not to coincide with the world (wrong bridge), (b) the addressee does not have enough information in order to

[1]These moves do not have the most intuitive surface form of CRs, namely question form, but instead they exhibit declarative form. Declarative form is in fact the most common realization of CRs found in corpora (see Section 1.2.4 for discussion).

infer the implicated premises (no bridge) or (c) according to the addressee's information, the command is ambiguous in ways that are relevant for the task (more than one bridge). However, none of these causes applies to this example, Frolog does have enough information in turn (7) in order to infer the unique bridge "stand up and take the key" which is executable in the game world (in particular, Frolog knows that standing up will make the key accessible, and that taking the key results in holding the key).[2].

Once these three potential causes have been discarded we are left with the "easy option": Frolog does not want to do its share of the task. I call this the easy option because this is the explanation that most frequently comes to mind when observing an interaction like this one between two interlocutors. In this interaction with Frolog, a feeling of non-cooperativity arises (as it did with my sister's example in Chapter 1), but clearly Frolog is not being uncooperative. The reason why Frolog is not bridging internally is not that he doesn't want to but that he does not have the necessary inferential capabilities. To use the terminology of Chapter 3, *Frolog cannot do practical inference*. If Frolog was able to do practical inference, using the information that is available to him in this sample interaction, he could infer that in order to unlock the door with the golden key all it needs to do is to stand up and take the key from the table. That is, Frolog enhanced with practical inference capabilities would react as illustrated in Figure 6.2. We include in Figure 6.2 the definitions and action specifications missing in Figure 6.1 and involved in the inference.

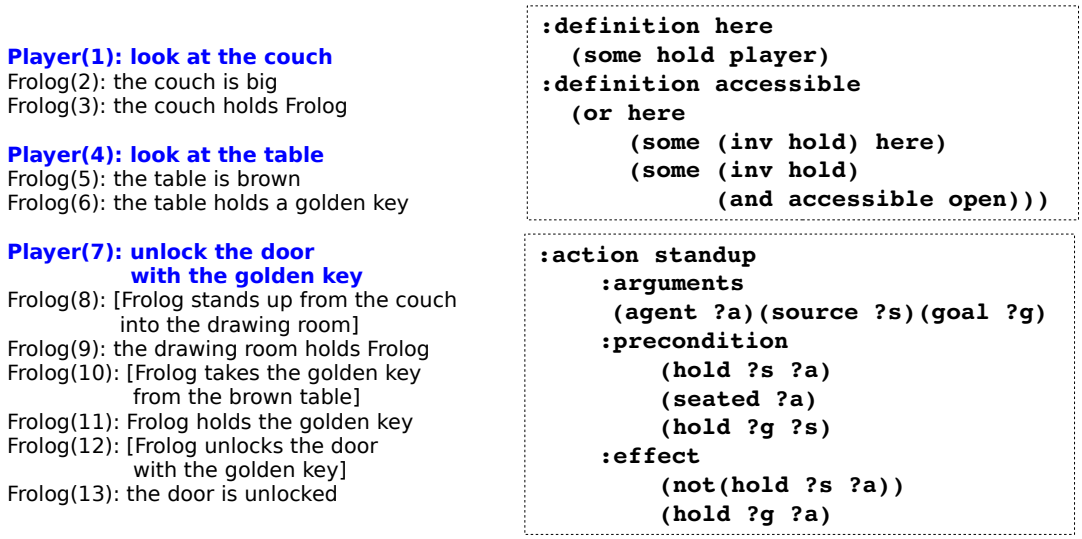


Figure 6.2: Frolog *with* bridging abilities

It is the work of this chapter then to explain how Frolog can come up with the appropriate implicated premises using the information in its knowledge bases. That is, in this chapter we extend the system presented in Chapter 5 with *bridging abilities*. The system that we will present is able: to *switch granularities* (in the terminology of Chapter 1), to *pose clarifications* (in the terminology of Chapter 2), to *do practical inference* (in the terminology of Chapter 3), and to *implicate collaboratively* (in the terminology of Chapter 4).

[2]You can verify this by looking at Frolog's action specification and ontology specification in Chapter 5

6.1.1 When does the process start inside Frolog?

When should the bridging process start inside Frolog? Let's answer this question in the light of what we learned in Chapter 4. In accommodation terms: if something was said (a command) that requires a component (a precondition) to have a certain value (to hold) in the conversational score (in the state of the game), then the component is accommodated. In bridging terms: if the listener does not find the antecedent of the given information (the precondition) directly in memory (in the state of the game), then the antecedent has to be bridged. Thus, in Frolog the bridging process starts when the player issues a command and some of its preconditions do not hold in the current state of the game.

Let's see how this applies to the command "Unlock the door with the golden key discussed above. When the player gives this instruction, it cannot be directly executed because the precondition `hold(frolog key1)` does not hold in the state of the game world. This is exactly the kind of situation that starts the bridging process. Now it is the case that the key is on the table so `hold(table1 key1)` holds and that the key can be in one place at a time. It is one of the internal ontological constraints of the game that the objects in the game world can be located in one place at a time. As a consequence, the precondition that is required cannot just be added (as a strict notion of accommodation would suggest, see Chapter 4) because this would cause a contradiction.[3] However, the precondition can be *made* true in the game scenario by *performing the appropriate actions*. This process can be seen as an implementation of the following generic pattern.

> *C is a condition that can be manipulated by an audience H. An agent S is observed by H to be doing A while C is mutually known to be false. H then acts to make C true, and S expects H to so act. [Thomason et al., 2006, p.36]*

According to Thomason *et al.*, inducing implicatures by **flouting** a rule of conversation, conforms to this generic pattern. In Frolog, C are the preconditions that can be manipulated by Frolog: the agent S is the player who gives an instruction that requires C while C is mutually known to be false. Frolog then must perform actions that make C true and the player expects Frolog to so act.

In our example, it is mutually known by the player and Frolog that Frolog is not holding the key; the key is on the table. However, the player asks Frolog to performs an action that requires Frolog to be holding the key, and expects Frolog to exploit their mutual knowledge to do whatever is necessary in order to follow the command. Hence, Frolog should act to make `hold(frolog key1)` true. And it does so by executing tacitly the actions `standup(frolog couch1 drawing-room)` and `take(frolog key1 table1)`.

The key question is then how can Frolog infer the 'appropriate' actions. And this is by no means an easy task. There are many types of information that come into play and interact. These sources can be used in different ways giving different results. To start with, not all failed

[3] In fact the contradiction is a result of the *unique name assumption* which we all assume implicitly (or did you think that Frolog was the table?) but needs to be made explicit to Frolog's knowledge management system.

preconditions are cases of flouting, for instance if the player utters the command "take the wall" Frolog can very well respond "I can't do that! The wall is not takeable", the player cannot reasonably expect Frolog to achieve a precondition that cannot be manipulated by Frolog. Moreover, if Frolog does not know where the golden key is, the player cannot expect Frolog to directly take it from wherever it is when she utters the command "unlock the door with the golden key". However, Frolog may reasonably look into all accessible drawers and chests (say) as a result of this same command.

So there are limits to the bridges that can be constructed. In Section 6.2 we will explore the construction of such bridges using classical planning. Classical planning will turn out to be too constrained for the kind of bridges that, even in Frolog's simple setup, need to be constructed. In Section 6.3 we then explore the construction of bridges using a less constrained kind of planning, namely planning with knowledge and sensing.

6.2 Bridging with classical planning capabilities

Remember (from chapter 3) that classical planning makes the assumption of complete information. We know that bridges are constructed to the mutual information which is, by no means, complete. The question underlying this section is then: in spite of this limitation, how well does classical planning work for coming up with the appropriate tacit acts.

In order to explore this question we first describe, in Section 6.2.1 how classical planning problems are generated on the fly each time bridging is required. Then, in Section 6.2.2 we explain how the solutions of such planning problems can be used to perform internal and external bridging. Finally, we explore the limitations of classical planning for the task of bridging. Interestingly, the most explored case of strict accommodation (and supposedly the easy case) turns out to be the most problematic here.

6.2.1 Specifying planning problems

In order to explore this question, the classical planner Blackbox [Kautz and Selman, 1999] was used. Blackbox, as any classical planner, takes three inputs (in PDDL): the initial state, the goal, and the available actions. Now, the question of 'what these three elements should contain' raises a number of subtle issues. Their discussion will highlight the kinds of problems that need to be considered when incomplete knowledge is handled under a complete information assumption.

The initial state

The first question is to decide the information that is needed for the initial state. In Frolog, two types of information are registered: complete and accurate information about the game world in the world KB and a representation of the common ground (constructed during the

interaction) in the interaction KB. Which of these should be used in order to discover tacit actions? In fact, we need both.

Let us analyze this decision by modifying our running example. Suppose that the golden key, which was lying on the table, was taken by a thief without Frolog and the player knowing. As a consequence, the key is on the table in the interaction KB, but in the world KB the thief has it. In this new scenario, the player issues the command "Unlock the door with the golden key." If we included in the initial state the complete information of the game KB, Frolog would automatically take the key from the thief (for example, by using the steal action) and unlock the door; but Frolog cannot possibly do this because Frolog does not know where the key actually is.

Let's try now with the interaction KB. In this case, Frolog would decide to take the key from the table and unlock the door with it. But this sequence of actions is not executable in the game world because the key is no longer accessible (the thief has it). That is, a sequence of tacit actions found by reasoning over the interaction KB might not be executable in the game world because the interaction KB may contain information that is inconsistent with respect to the world KB. Hence, we need both KBs: we infer the actions intended by the player using the information in the interaction KB but we have to verify this sequence of actions on the world KB to check if it can actually be executed.

The problem is, more generally, that a sequence of tacit actions found by reasoning over the interaction KB might not be executable in the game world because the interaction KB contains incomplete information but the planner is assuming that the information is complete. That is, the planner is applying the closed world assumption on the interaction KB: the planner assumes that each unknown proposition is false. This is certainly far from ideal, but is as good as things can get with an assumption of complete information. As it turns out, however, the effects of this limitation are not too dramatic because each plan found is then verified for executability on the complete and accurate information of the world KB, so such incorrect plans will be discarded.

Summing up, the **action inference** step is done using the planning services provided by BLACKBOX on the mutual information contained in the interaction KB at the moment in which the instruction that is being bridged was uttered. The **action executability** step is done on the complete information (using the reasoning services provided by RACERPRO in the usual way).

The goal

Let us now define what the goal of the planning problem should be. Frolog should act to make the preconditions of the action true with one restriction. The restriction is that it must be possible for Frolog to manipulate these preconditions. In principle, we shouldn't worry about this restriction because the planner should take care which propositions are manipulable by Frolog and which are not, given the current state. So we could just define the goal as the conjunction of all the preconditions of the command uttered by the player. For example, when

the player says "Unlock the door with the key" the goal of the planning problem will only include the atoms:

```
locked(door1),
hold(frolog key1),
accessible(door1),
fitsin(key1 door1)
```

However, here again the incomplete information limitation comes into play. If Frolog does not know that the golden key fits into the chest then the planner will not be able to find a plan for the previous goal, because no action modifies the predicate `fitsin`.

The workaround that we found for this problem is not to include in the goal those literals which contain static predicates. A **static predicate** [Nau *et al.*, 2004] is a predicate which is not changed by any action in the planning domain. Thus in a planning problem, the true and false instances of a static predicate will always be precisely those listed in the initial state specification of the problem definition. Note that there is no syntactic difference between static and dynamic predicates in PDDL: they look exactly the same in the `:predicates` declaration part of the domain. Nevertheless, some planners automatically calculate them from the planing domain specification and support different constructs around static and dynamic predicates, for example allowing static predicates (but not dynamic ones) to be negated in action preconditions.

Restricting the goal to those predicates that are not static, when the player says "Unlock the door with the key", the goal of the planning problem will only include the atoms:

```
locked(door1),
hold(frolog key1),
accessible(door1)
```

The literal `fitsin(key1 door1)` is not included in the goal because the predicate `fitsin` is static from a classical planning perspective where the information is complete. A classical planning perspective models the state of the world and then, if a given key does not fit into a given lock, it is not possible to make it fit. However this is not as straightforward as it seems; which predicates are static turns out to be a subtle issue in our framework. This is due to the fact that the model to which we are applying planning, namely the interaction KB does not model complete information about the state of the world but rather incomplete knowledge on this world. So in this KB the `fitsin` should not be static: if Frolog doesn't know whether the key fits into the lock it is indeed possible to find out (he can try the key on the lock). From a planning on knowledge information perspective the predicate `fitsin` can be modelled as not static (we explore these issue in Section 6.3)

The actions

To complete the picture, the actions available to the planner are all the actions in the game action database. This means that we are assuming that all the actions schemes that can be executed, such as the one reproduced below, are mutually known to Frolog and the player.

```
:action unlock
      :arguments (agent ?w) (theme ?x) (inst ?y)
      :preconditions
         (accessible ?x)
         (locked ?x)
         (fitsin ?y ?x)
         (hold ?w ?y)
      :effects
         (not(locked ?x))
         (unlocked ?x)
```

In order to be able to perform bridging to the mutual information it must be mutually known what the preconditions and the effects of the actions involved are. In our unlock the door example, if the player doesn't know that the agent needs to be holding the key in order to unlock the door with it, then the player cannot possibly implicate that Frolog should take the key by saying "Unlock the door with the golden key". The same goes *mutatis mutandis* for Frolog.

The assumption that the player and Frolog know the exact specification of all the actions that can be executed in the game world is a simplifying assumption. In the framework designed in Chapter 4 we specified that the understanding of the actions of one of the dialogue participants did not need to coincide with that of the other, or with the actual behavior of the actions in the real world. In Frolog we make this simplifying assumption not because we cannot use two different action databases for these two purposes but because if we did we would then need to decide (and implement) how these two actions get coordinated. This cannot be avoided because as soon as such specifications are mis-coordinated people coordinate them through dialogue and the truth is that this is a complex process which is far from being well understood [DeVault *et al.*, 2006; Mills, 2007; Larsson, 2007]. In Chapter 4 this was not a problem because we just needed a 'picture' of the current state of both databases but here we would need to automatize this coordination process inside the interactive system that Frolog is and, as we said, this is simply too difficult at present.

Implementation details

This section address the technical issue of how the bridge is reinserted into the Frolog control cycle. Remember from Chapter 5 that an interpretation of an instruction is a list of readings. Bridging starts when none of these readings is directly executable. For each failed reading Frolog tries to find a *sequence of actions* (i.e., a *bridge*) which transforms the current game scenario into a scenario where the reading can succeed. If no such bridge exists, the reading is discarded, otherwise the bridge is concatenated before the reading, enlarging the original sequence of actions. The new list of readings built in this way is reinserted into the action handling module and its execution proceeds as usual.

In order to illustrate the previous behavior of Frolog, let us consider again the command

"Unlock the door with the key" which in fact has two readings in Frolog (because of the syntactic ambiguity). One of the readings fails because there is no "door with the key" in the current game scenario. The other reading cannot be directly executed because Frolog is not holding the key. We know that for this reading the plan "stand up and take the key" is found. This plan is concatenated before the original reading and the extended reading is processed again by the action handling module. This time, the input of the action module will be the sequence of actions "stand up, take the key and unlock the chest with it", making explicit the tacit actions.

In order to infer tacit actions, Frolog uses the planning services provided by the planner Blackbox [Kautz and Selman, 1999]. Blackbox works by fixing the length of the plan in advance and iteratively deepening it. This behavior makes it particularly well suited for our needs because it finds optimal plans (minimal in the number of actions) and does it fast. Fast responses are essential for a natural interaction with the player. For a detailed description of the performance of Blackbox in Frolog see [Benotti, 2006a; Benotti, 2006b]. Moreover, optimal plans are crucial, otherwise actions which are executable in the game scenario but completely irrelevant to the player command might be included as tacit actions. For example, a non-optimal planner might not only take the key as in our example, but also take and drop other arbitrary objects as well.

6.2.2 Internal and external bridging in Frolog

In this section we present several examples of how the solutions of the planning problems introduced in the previous section can be used in order to perform internal bridging and trigger external bridging when appropriate.

In the text-adventure engine, Frolog's bridging ability can be activated or deactivated at will, making it possible to compare its behavior.

In the example in Figure 6.3, the player issued the command *put the green frog on the table* when Frolog is sitting on the couch. The action *put* requires its *goal* parameter, namely *the table*, to be accessible and this precondition does not hold in the game world. When bridging is deactivated Frolog will answer as it used to answer when it didn't have bridging abilities, verbalizing the failed precondition. But as soon as the bridging abilities are activated (in exactly the same state of the game) the same command is bridged and executed by Frolog. Frolog makes explicit its bridging process by verbalizing each action and its effects in order, which can become quite repetitive. Of course, this logorrhea can be restricted by not forcing Frolog to make its internal bridging reasoning explicit; however this is done on purpose in order to monitor Frolog's internal reasoning.[4]

The planning problems are generated on the fly during a game each time a precondition does not hold. For instance, the state to which the command "put the green frog on the table" is bridged to is different than the state to which the command "eat the yellow apple" is bridged to (in this last state the green frog is by the yellow apple on the table).

[4]Also, such logorrhea makes explicit the fact that Frolog *is* doing the bridging; users of dialogue systems don't expect a dialogue system to have such abilities.

(bridging dis-activated)

Player(1): put the green frog on the table
Frolog(2): Frolog can't do that!
The table is not accessible

(bridging activated)

Player(3): put the green frog on the table
Frolog(4): [Frolog stands up from the couch into the drawing-room]
Frolog(5): the drawing room holds Frolog
Frolog(6): [Frolog puts the green frog on the table]
Frolog(7): the table hold the green frog

Player(8): look at the table
Frolog(9): the table is red
Frolog(10): the table holds the green frog, the yellow apple and the key

Player(11): eat the yellow apple
Frolog(12): [Frolog takes the yellow apple from the table]
Frolog(13): Frolog holds the yellow apple
Frolog(14): [Frolog eats the yellow apple]
Frolog(15): the yellow apple is gone

```
:action put
    :arguments
     (agent ?w)(theme ?x)(goal ?y)
    :precondition
        (hold ?w ?x)
        (container ?y)
        (accessible ?y)
    :effect
        (not(hold ?w ?x))
        (hold ?y ?x)
```

```
:action eat
    :arguments
       (ingestor ?p)(ingestible ?x)
    :precondition
       (edible ?x)
       (not(disgusting ?x))
       (hold ?p ?x)
    :effect
       (not(hold ?p ?x))
       (gone ?x)
```

Figure 6.3: *Internal* bridging in Frolog

Bridges are inferred using the information in the interaction KB and their execution is verified with respect to the accurate and complete state of the world (the world KB). So Frolog distinguishes two ways in which accommodation can fail: there is no bridge and there is a bridge which is not executable in the game world. These two cases, plus the case in which there is more than one bridge, are the three potential causes of CRs that we modelled in our framework in Chapter 4. Frolog cannot integrate the analysis of multiple bridges (i.e., multiple plans) because current classical planners only return one plan. We illustrated the case of a bridge which is not executable in Section 6.2.1 when discussing the content of the initial state of the planning problems. Here we illustrate the case of no bridge with the interaction illustrated in Figure 6.4 (numbering of the turns starts in (16) because it continues the previous sample interaction). Frolog just entered a room called the treasury which has a white wall and the player issued the command (20) "take the wall". In this case, the precondition that fails, namely `(takeable wall1)` contains the static predicate `takeable` so the goal does not contain it. Thus the goal contains propositions that are already true in the initial state and then the planner will find an empty plan. As usual this sequence of actions, now empty, will be concatenated before the actions made explicit in the command and an attempt to execute them on the game world will take place. As expected, the command, which is unchanged, will fail again and Frolog will verbalize the failed static precondition.[5]

Instruction (29) is an example of an instruction that cannot be internally bridged but whose

[5]This failed precondition verbalization could be considered a triggering of external bridging if the the assumption of the completely synchronized action schemes is dropped (maybe the player knows how to make the wall takeable).

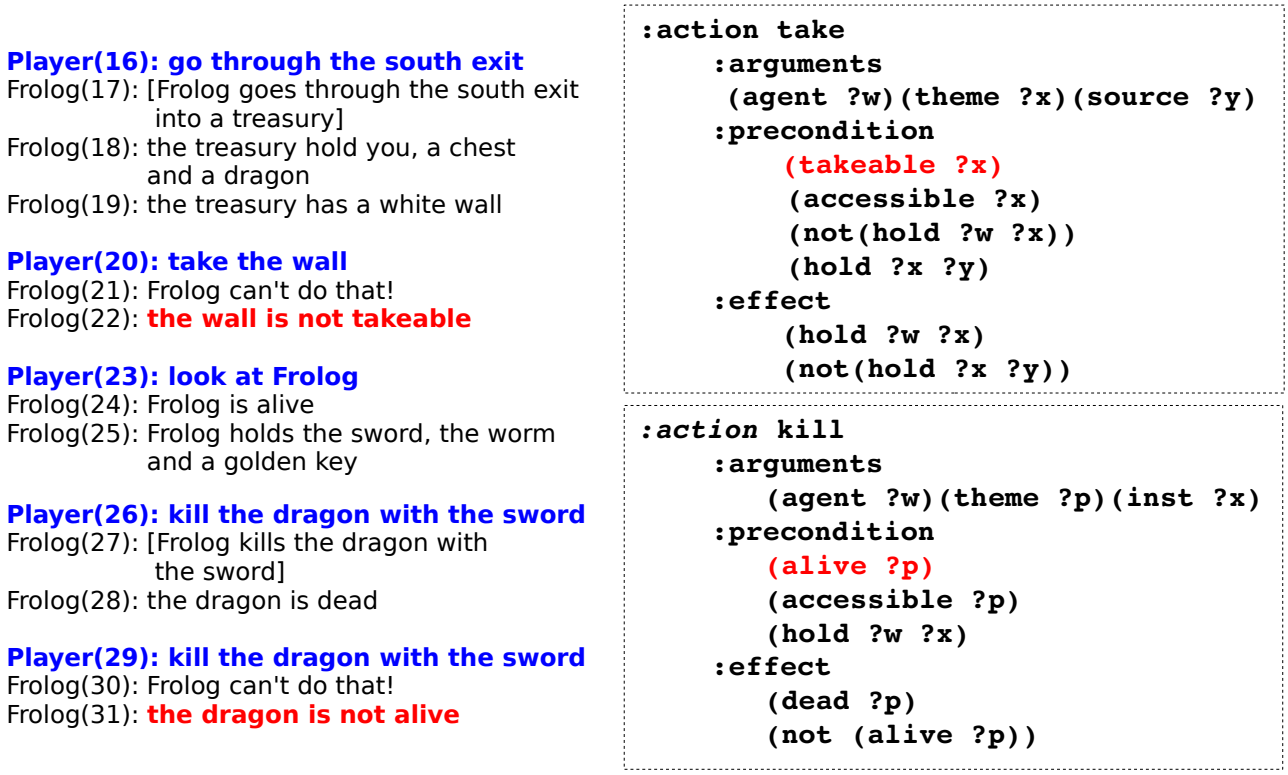


Figure 6.4: *External* bridging in Frolog (world constraints)

failed precondition is not on a static predicate. In this case, the planner will be invoked with the goal `(alive dragon1)` and the planner will not be able to find a plan: the dragon cannot be revived. Whenever the planner says there is no plan for all the preconditions in the goal, the planner will be invoked with each precondition separately. Certainly, for at least one of these preconditions no plan will be found, the first such precondition found will be verbalized, as in turn (31): *the dragon is not alive*.

Now, it is not only when the predicate is static or the actions are irreversible that the planner will find no plan. There is also the case in which Frolog does not have enough knowledge yet in order to infer the bridge.

This is the case in instruction (34) in Figure 6.5. In this instruction the player wants Frolog to open the chest but Frolog does not know how to do this. In the screen-shot in the right you can see state of the interaction KB from turns (32) to (40). In this model we can see that the player does not have information about what fits into the chest, in order to unlock it. This is why the planner is not able to find a plan that achieves the precondition `(unlocked chest1)` of the command "open the chest" in instruction (34). The instruction in (39) fails as well making the problem explicit. See Section 5.3.3 for explanation on how the negative feedback in utterance (40) is generated. Finally the player makes explicit that she wants Frolog to use the golden key to unlock the chest in (41) and this command is executed by Frolog in turn (42).

Frolog's behavior continues as expected until turn (48) in which Frolog again is not able to bridge the command open the chest. It is to be expected that the fact that the golden key fits into the chest is mutual knowledge by now but Frolog is not reacting accordingly.

The problem is that in the architecture that we have designed using classical planning, the mutual knowledge can only be updated with information in the effects of the physical actions that are possible

Player(32): look at the chest
Frolog(33): the chest is closed and
locked and wooden

Player(34): look at Frolog
Frolog(35): Frolog is alive
Frolog(36): Frolog holds the sword, the worm
and the golden key

Player(37): open the chest
Frolog(38): Frolog can't do that!
the chest is not unlocked

Player(39): unlock the chest
Frolog(40): Frolog can't do that!
Frolog doesn't know what fits into the chest

Player(41): unlock the chest with the golden key
Frolog(42): [Frolog unlocks the chest]
Frolog(43): the chest is unlocked

Player(44): lock the chest with the key
Frolog(45): [Frolog locks the chest]
Frolog(46): the chest is locked

Player(47): open the chest
Frolog(48): Frolog can't do that!
the chest is not unlocked

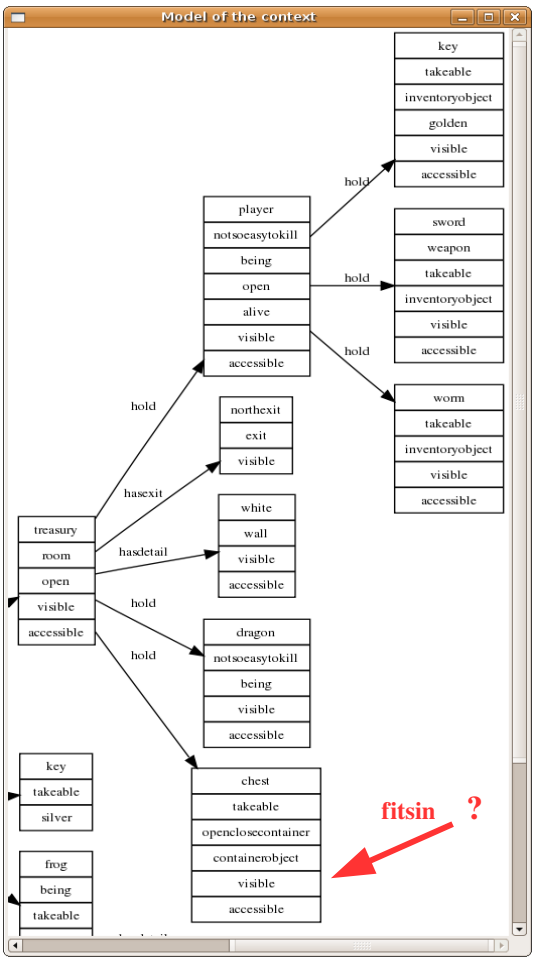


Figure 6.5: *External* bridging in Frolog (knowledge constraints)

in the game world. And there is no physical action that achieves `(fitsin key1 chest1)` so there is no way in this setup that this kind of information gets into the mutual knowledge.

In order to overcome this issue there are two options: (1) either we allow different sources to update the mutual information loosing uniformity in the update process or (2) we add actions that manipulate this kind of information. We describe option (1) that in its straightforward cases turns out to be the procedural version of "strict accommodation", and when generalized is of a piece with abduction. Once the limitations of (1) are analyzed we explore option (2) which requires moving into non-deterministic planning; we present this solution in Section 6.3.

We could say that the intuition behind the strict notion of accommodation is that when the speaker utters a sentence, she is not only trying to achieve the obvious effects of the utterance, but is also communicating the ways in which she assumes the world to be, and on which the success of the utterance depends. And then the requisites of the utterance will "come into existence".

Following this intuition it can be argued that the intention behind the utterance "unlock the chest with the golden key" in turn (41) is twofold. It is clear that the player wants to achieve the effects of the action, that is, she wants to have the chest unlocked. But the player also expects Frolog to recognize the assumptions she is making and on which the success of the utterance depends. In particular, she expects Frolog to recognize that she is assuming that the golden key fits into the chest lock (this is why Frolog can coherently ask "why do you think that the golden key will fit?").

In particular, after the action (`unlock(door1 key1)`) is executed successfully, the player will

believe that the door is accessible and that the golden key is in Frolog hands, facts that she already knew. However, she will also believe that the door is now unlocked, the obvious effect of the action; and that the golden key fits in the door lock, the assumption she made and was confirmed by the success of the action. This means that, when an action is executed, the interaction KB will be updated not only with the effects of the action but also with its preconditions. And those preconditions that were not in the interaction KB before will be verbalized as in turn (9) in Figure 6.6.

When performing this update, the order in which the changes are made is important in order to leave the KB in the intended state. Concretely, the KB should be first updated with the player preconditions and then with the player effects. Otherwise, the preconditions might undo the effects of the action. This is in line with the intuition that preconditions come before than effects.

Player(1): unlock the chest
Frolog(2): Frolog can't do that!
Frolog doesn't know what fits-into the chest

Player(3): look at you
Frolog(4): you are alive
Frolog(5): you hold a golden key

Player(6): unlock the chest with the golden key
Frolog(7): [Frolog unlocks the chest with the golden key]
Frolog(8): the chest is unlocked
Frolog(9): **the key fits-into the chest**

Player(10): lock the chest
Frolog(11): [Frolog locks the chest with the golden key]
Frolog(12): the chest is locked

Player(13): look at the chest
Frolog(14): the chest is closed and locked and wooden

Player(15): open the chest
Frolog(16): [Frolog unlocks the chest with the golden key]
Frolog(17): the chest is unlocked
Frolog(18): [Frolog opens the chest]
Frolog(19): the chest is open
Frolog(20): the chest hold a golden crown

```
:action unlock
      :arguments
      (agent ?w) (theme ?x) (inst ?y)
      :precondition
          (accessible ?x)
          (locked ?x)
          (fitsin ?y ?x)
          (hold ?w ?y)
      :effect
          (not(locked ?x))
          (unlocked ?x)
```

```
:action open
      :arguments
      (agent ?w) (theme ?x)
      :precondition
          (accessible ?x)
          (closed ?x)
          (unlocked ?x)
      :effect
          (not(closed ?x))
          (open ?x)
```

Figure 6.6: *External* bridging becomes *internal* as common ground accummulates

The rest of the interaction (from turns (10) to (15)) show that once the proposition `(fitsin key1 chest1)` is added to the interaction KB the action "open the chest" can be internally bridged even when the chest is locked. Because the player and Frolog mutually know which key fits into the chest. This proposition "came into existence" because its a requirement of an action that Frolog executed, and since the action was executed successfully then the proposition must hold.

At first sight, this seems a good strategy for adding world information to the mutual knowledge. However, we have to be careful: this world information is only added at execution time; with this change, the execution process and the bridging process become mis-coordinated. That is, when executing the unlock action, Frolog will now acquire the knowledge of whether the key used fits or not. But he cannot reason over this information acquisition process: he cannot *decide* to try the action

unlock in order to learn whether the key fits or not. For example, if there are several keys available and the player tells Frolog "open the chest" (when they don't know which key fits) Frolog cannot possibly infer that it is reasonable to try all the keys. With this strategy of update of the mutual knowledge, the actions have more effects during execution than during bridging.

If we want to mimic what now is going on during execution during bridging we need to modify planning; in particular the complete knowledge assumption has to be dropped. That is, propositions that are not known in the interaction KB (such as `(fitsin key1 chest1)`) cannot be taken to be false (by applying the complete knowledge assumption) but can be assumed when needed by a precondition. As a result planning is constrained only by those propositions that are known in the interaction KB, the propositions that are unknown can be added to the initial state as required. That is, the initial state is constructed on the fly. But planning modified in this way is of a piece with abduction and with its computational complexity.

Fortunately, there is also a more restricted alternative that we referred to as (2) in Subsection 6.2.2. That is, adding actions that explicitly state in which conditions certain information can be acquired. These actions are called **sensing actions** in the literature [Levesque, 1996]. We spell out how bridging can be extended with reasoning on sensing acts in the next Section.

6.3 Bridging with planning and sensing

In this Section we add to our model the treatment of sensing actions.[6] To this end, we have integrated a planner that is able to find plans in the presence of incomplete knowledge and sensing, namely PKS [Petrick and Bacchus, 2002] into Frolog. In the resulting model, we study the interactions among dialogue acts, physical acts and sensing acts.

6.3.1 Why is sensing necessary?

When interlocutors are engaged in situated dialogue, it is evident that their informational states evolve as a result of the dialogue acts performed during the task, and through the physical acts that interlocutors perform on their environment. But their states are also updated with the information that the participants sense from their environment; embedded agents do not have complete information about the world but they can sense it. Before execution, a sensing act has an outcome that is unpredictable; thus we say then that a sensing act then is a **non-deterministic act**.

Bridging and grounding of dialogue and physical acts are topics that have been widely studied (see [Traum, 1994; DeVault, 2008] for paradigmatic computational models). But the study of accommodation and grounding of sensing acts is also essential when agents are embedded and can sense their environment. Moreover, sensing acts are usually less evident than physical and dialogue acts. Hence, an important question to study is "When is the common ground of the dialogue updated with the sensed information?" Or in other words, "When is there in the state of the activity enough evidence that a piece of information has been sensed?" Let us address these questions with an example:

In kindergarten, the teacher showed a green square to a boy and, offering a piece of paper, told him: "Paint a circle that has this same color".

[6] The initial ideas behind this section were published in [Benotti, 2008b].

This simple example illustrates the interaction of a *dialogue act* performed by the teacher (request) with a *sensing action* (sense color) and a *physical action* (paint) that the teacher expects from the boy. When giving this instruction the teacher relied on the ability of the boy to sense the colors, but the sensing action is left tacit in the teacher request. She could have make it explicit saying "Look at the color of the square and paint a circle that has the same color". However in conversation, sensing actions are more naturally left tacit than made explicit. Why? Because they are so natural for sensing agents (indeed, sometimes they are unavoidable) that it is extremely easy to take them for granted.

To analyze how this bridging process starts, we are going to look at this example as an instance of the general *rule of accommodation* introduced by Lewis:

> *If at time t something is said that requires component s_n of conversational score to have a value in the range r if what is said is to be true, or otherwise acceptable; and if s_n does not have a value in the range r just before t; and if such and such further conditions hold; then at t the score-component s_n takes some value in the range r. [Lewis, 1979, p.347]*

Bearing this schema in mind, let us analyze step by step the different values that the variables of the rule take for our simple example. First of all, what's t? This is what Stalnaker has to say here:

> *The prior context that is relevant to the interpretation of a speech act is the context as it is changed by the fact that the speech act was made, but prior to the acceptance or rejection of the speech act. [Stalnaker, 1998, p.8]*

So in our example t is the time right after the teacher said "Paint a circle that has this same color" but before the acceptance or rejection of this request.

Now, let us determine what the relevant components s_n are. Suppose that the boy is color blind and the teacher knows it. Then her request does not make much sense and any side participant and the boy himself will start asking what the goal of the request is, because clearly it cannot be the literal one (to obtain a green circle). Therefore, the color referred to by the teacher is the s_1 of our example. And if what the teacher said is to be acceptable, s_1 is required to have a particular value r_1; the same color than the square has in the real world (or in fact, a representation of it). Furthermore, there is no evidence that s_1 already has the value r_1 before the teacher began to speak (that is, there is no evidence that the color has been under discussion before), so we can assume that it doesn't.

Now, what are the further conditions that need to hold so that, at t, the score-component s_1 takes some value r_1? The teacher and the boy both know (at least intuitively) that people can sense their environment, that members of the same culture usually assign the same name to the same parts of the spectrum of colors, that humans can remember facts that they sense, that the sensed object is accessible, that a person will actually sense the color of an object if he is required to know this fact; the teacher and the boy rely on these and many other things that are usually taken for granted. All this knowledge is necessary for the boy to come up with the right sequence of actions in order to respond to the teacher's request; that is, in order to sense the color of the square and paint the circle.

Following Lewis, we would finish our instantiation of the rule of accommodation with the fact that at the time t the score-component s_1 takes value r_1. Two last comments are in order here. First, it is worth pointing out that at the moment t the request has not yet been accepted or rejected but the addressee has already taken it in and adjusted himself to the fact that the dialogue act has been

performed. The acceptance or rejection can be seen as a second change to the conversational record that occurs after the rule of accommodation applies. It's very important to distinguish between these two changes. Why? Because even if the request is rejected, the update of the conversational record that resulted from the accommodation may remain. Even if the boy answers "I don't like green. I won't do it", we know that the boy sensed the color of the square.

Second, how does the score-component s_1 takes value r_1? Theories that adopt a strict notion of accommodation seems to suggest is that s_1 takes value r_1 and nothing else changes. However, if this is the case, how can we explain the fact that the boy may take off a blindfold (he was playing "Blind man's bluff") after hearing the teacher? The sensing acts can also have their requirements (or preconditions) and side-effects, and we think that a natural way to model the accommodation updates is through **tacit sensing acts**. As studied in previous work, physical acts can be left tacit [Thomason *et al.*, 2006; Benotti, 2007], dialogue acts can be left tacit [Kreutel and Matheson, 2003; Benotti, 2008a], but also sensing acts can be left tacit.

The analysis of our example so far has given us some insight on the questions that were raised in the beginning of this section. We have seen that tacit sensing can be grounded even if the dialogue act that required the sensing is directly rejected (the "boy doesn't like green" example). And it can also be the case that the tacit sensing is grounded even if it cannot be directly executed because, for instance, it requires the execution of some physical act first (the "Blind man's bluff" example). The interactions among sensing acts, dialogue acts and physical acts can be extremely subtle. Modelling them inside Frolog (putting sensing, physical and dialogue acts in a common schema) and, in particular making explicit the information at play, is the topic of the rest of this section.

Subsection 6.3.2 describes the main features of the planner that we use for our formalization and implementation, and then briefly presents the kinds of sensing acts that ocurr in Frolog. In subsection 6.3.3 we explain in detail how a command issued by the player that includes tacit sensing actions is interpreted using the planner, and then executed by the game

6.3.2 Specifying the planning problems

PKS [Petrick and Bacchus, 2002; Petrick and Bacchus, 2004] is a knowledge-based planner that is able to construct conditional plans in the presence of incomplete knowledge. PKS builds plans by reasoning about the effects of actions on an agent's knowledge state. In PKS, the agent's knowledge state is represented by a set of databases, where each database represents a different kind of knowledge. Actions are represented as updates to these databases and, thus, describe modifications to the agent's knowledge state, rather than physical updates to the world state. The four databases used are as follows:

K_f database: The first database is much like a standard STRIPS database except that both positive and negative facts are allowed, and the closed world assumption does not apply. Each ground literal $l \in K_f$ means that the planner knows that the agent *knows that* l is true.[7]

[7]Unlike STRIPS, the FOL language used in K_f can contain functions, but in a restricted way. All the terms that appear in any literal must be constants. K_f allows formulas of the form $f(c_1, \ldots, c_n) = c_{n+1}$ or $f(c_1, \ldots, c_n) \neq c_{n+1}$, where f is an n-ary function and the c_i are all constants. In effect, this restriction means that function values in K_f are considered to be known by the agent only if they can be grounded out as constant values.

K_w database: The second database is designed to represent the knowledge effects of sensing actions. Intuitively, $l \in K_w$ means that the agent *knows whether* l; that is, the planner knows that the agent knows that l is true or that l is false, but does not know which. K_w can contain any formula that is a conjunction of atomic formulas.

Remember from chapter 3 that the difference between K_f and K_w is what resulted in non deterministic acts being included into a plan. As discussed in Chapter 3, a binary sensing act, such as "you look outside the window", adds to K_w the proposition that says that *you know whether* it is raining or not. Hence, from my point of view, your action you look outside can leave my K_f database in any of two states, either $\neg$raining $\in K_f$ or raining $\in K_f$. That is to say, and let me repeat *from my point of view*, the action you look outside is non-deterministic (see Figure 6.7 from Chapter 3 reproduced here); the action you look outside is a binary sensing act.

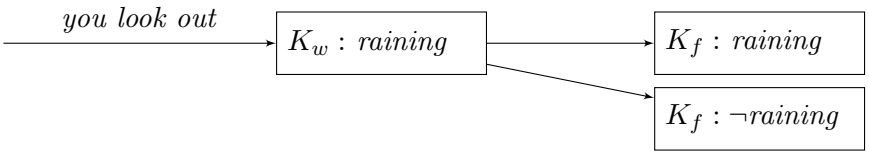


Figure 6.7: A non-deterministic action results in two states

Binary sensing acts result in **conditional plans** to be built (that is, plans that are not sequential but contain branches). A case-study of the use of conditional plans inside Frolog is presented in the next section.

K_v database: The third database is a specialized version of K_w designed to store information about certain function values the agent will come to know when executing sensing actions. In particular, K_v can contain any unnested function term. For example, $f(x, c)$ is a legal entry in K_v but $f(g(c), d)$ is not. Like K_w, the entries in K_v are used to model sensing actions, except in this case K_v is used for the modeling of sensing actions that return constants (e.g., numeric values), rather than truth values.

Since the planner does not know which values a certain function term $f \in K_v$ can take, the sensing actions that sense f result not in conditional plans but in plans with **runtime variables** [Levesque, 1996]. Plans that contain runtime variables are not ground; that is, they contain a variable. A case-study of the use of plans with run-time variables will be presented in next Section.

K_x database: The fourth database contains information about a particular type of disjunctive knowledge, namely **exclusive-or** knowledge of literals. Entries in K_x are of the form $(l_1|l_2|\ldots|l_n)$ where each l_i is a ground literal (ground functional equalities are also permitted). Intuitively, such a formula represents knowledge of the fact that exactly one of the l_i is true. Hence, if one of these literals becomes true, all the other literals are immediately false. Similarly, if $n-1$ of the literals become false then the remaining literal must be true.

This database does not introduce a new type of sensing action, nor a new type of resulting plan. However, it can be used to prune the search space in a consistent way in combination with K_w and K_v (we will see an example of this in Section 6.3.3). If the planner has K_v knowledge of a function, it can use functional K_x knowledge to insert a multi-way conditional branch into a plan. Along each branch the planner will add to K_f the assumption that the function is equal to one of its possible mappings, then try to continue constructing the plan using this function mapping. For instance, if $g(x)$ has

the range $(d_1, d_2, \ldots, d_m)$ and the planner also has Kv knowledge of $g(c)$, then it can construct an m-way branch into a plan; along branch i, the planner would add the assumption $g(c) = d_i$ to K_f and continue planning with that knowledge.

To sum up, the content of the four databases constitute the state of the planner. K_f is like a standard STRIPS database except that both positive and negative facts are stored and the closed world assumption does not apply. K_v stores information about sensing actions that return numeric values. K_w models the effects of binary sensing actions that sense the truth value of a proposition. K_x models the agent's exclusive disjunctive knowledge of literals (that is, the agent knows that exactly one literal from a set is true).

In the prototype we have implemented using PKS inside our text-adventure game, PKS response time was acceptable (less than 2 seconds) for the kind of planning problems that the text adventure typically gives rise to. We tested it using the breadth first search strategy, rather than depth first because we require optimal length plans.

There are two sorts of sensing actions, corresponding to the two ways an agent can gather information about the world at run-time. On the one hand, a sensing action can observe the truth value of a proposition $P(c)$, resulting in a conditional plan. The kind of incomplete knowledge sensed by this kind of action can be described as *binary* because it represents the fact that the agent knows which of the two disjuncts in $P(c) \vee \neg P(c)$ is true. In PKS, binary sensing actions are those that modify the K_w knowledge base. On the other hand, a sensing action can identify an object that has a particular property, resulting in a plan that contains run-time variables. The kind of incomplete knowledge sensed by these kind of action can be described as *existential* because it represents the fact that the agent knows a witness for $\exists x.P(x)$. In PKS, existential sensing actions are those that modify the K_v database.

6.3.3 Bridging sensing acts in Frolog

In this section we present one case study of the treatment in Frolog of each of the kinds of sensing acts discussed in the previous section. First we present a case-study of disjunctive knowledge that makes use of conditional plans. And then, we describe a case-study of existential knowledge that makes use of parametric plans.

Tacit actions in conditional plans

We are going to analyze commands issued by the player that involve the execution of binary sensing actions that result in conditional plans.

In order to motivate conditional plans, let us consider an example. Suppose that the player is in a room with a locked door. She is looking around searching for a way to open the door, when Frolog says that there are two keys (one silver and one golden) lying on a table in front of it. Then she inputs the command "Open the door". Given the current state of the game, it would be reasonable for Frolog to execute the following conditional plan. The plan involves taking both keys, trying the silver one in the door, and (if it fits) unlocking and opening the door; otherwise the golden key is used. This plan is reasonable, however it is not guaranteed that it will succeed: it can be that none of the keys fits into the door lock.

```
<init>
  take(silver_key,table)
  take(golden_key,table)
  trykey(silver_key,door)
  <branch,fits_in(silver_key,door)>
  <k+>:
     unlock(door,silver_key)
     open(door)
  <k->:
     unlock(door,golden_key)
     open(door)
```

Should Frolog execute this plan for the player? Or in more general terms, when can this command (which gives rise to particular implicatures) be bridged internally? This depends on what has already happened in the game. Has the player already been through enough experiences to have the knowledge that is necessary in order to "open the door"? If yes, don't force the player to repeat the boring steps.

But how can we represent the knowledge that is necessary in order to find the conditional plan involved by this command, in order to leave the necessary actions tacit? To illustrate our explanation, let us go back to the concrete input "Open the door" and its conditional plan and analyze how it is handled by the system. The sensing action involved in the conditional plan is `trykey` defined in PKS as follows:

```
<action name="trykey">
  <params>?x, ?y</params>
  <preconds>
    Kf(accessible(?x)) ^
    Kf(locked(?x)) ^
    Kf(key(?y)) ^
    Kf(inventory_object(?y))
  </preconds>
  <effects>
    add(Kw, fits_in(?y,?x));
  </effects>
  </action>
```

Intuitively, after executing the action `trykey(?x,?y)` the agent *knows whether* a particular key `?x` fits in a locked object `?y` or not. Is this knowledge enough to find the conditional plan above? No, because it could be the case that none of the two keys fits into the door. If this is a possibility, then the conditional plan may not achieve the goal `Kf(open(door))`. In order to rule out this possibility the following facts have to the added to the initial state of the planning problem:

```
  add(Kx, fits_in(k1,c1)|fits_in(k2,c1))
```

Given this information, PKS is able to come up with the conditional plan above. In its current version, PKS only returns disjunctive plans that will always be successful given the specification of

the planning problem. It doesn't matter what the actual configuration of the world is, PKS guaranties that there will be a branch in the plan that achieves the goal. If this cannot be achieved then PKS will say that there is no plan. However, it might be the case that there is some conditional plan that is successful for most but not all configurations of the world. It would be interesting to have a planner that could provide plans for these cases, even when some of the branches will not achieve the goal.

Implementation details

Conditional plans are executed by decomposing them in disjunctive plans. For example, the conditional plan shown above can be decomposed in two disjunctive plans, namely:

```
take(silver_key,table)
take(golden_key,table)
unlock(door,silver_key)
open(door)
```

and

```
take(silver_key,table)
take(golden_key,table)
unlock(door,golden_key)
open(door)
```

These two disjunctive plans can be directly inserted in the game flow. In the game, the semantic representation of a command is in disjunctive normal form (that is, it is a disjunction of conjunction of actions). Each disjunct corresponds to a different reading of the command, hence a command's semantic representation will contain more than one disjunct if the command is ambiguous. Here, each branch of the plan can be reinserted into the game flow as a disjunct in the semantic representation of the command. Only one of the branches will be successfully executed since the sensed information is known to be exclusive (only one of the keys fits).

Run-time variables in tacit actions

In this section we are going to analyze commands issued by the player that involve the execution of existential sensing actions. Existential sensing actions result in parametric plans, that is, plans that include actions with run-time variables, values that will only be known at run time.

Frolog and the player found a room with a panel where the location of all individuals in the game world can be checked. The player and Frolog know that in this game scenario, Frolog can drive itself to any other location if it knows the destination, and that in order to kill someone you have to be in the same place. The player wants Bill dead and so she utters the command "Kill Bill". How do we have to represent this information so that the planner will be able to come up with a successful plan? The goal of the command can be represented with `Kf(dead(bill))` and the information about how the game world works that is already available to the player and Frolog can be represented with the following action schemes:

```
<action name="checklocation">
  <params>?x</params>
  <preconds>
    Kf(player(?x))
  </preconds>
  <effects>
    add(Kv, haslocation(?x));
  </effects>
</action>
<action name="drive">
  <params>?x,?y</params>
  <preconds>
    Kf(player(?x)) ^
    Kv(?y) ^
    Kf(haslocation(?x)!=?y)
  </preconds>
  <effects>
    add(Kf, haslocation(?x)=?y);
  </effects>
</action>
<action name="kill">
  <params>?x, ?y</params>
  <preconds>
    Kf(player(?x)) ^
    Kf(player(?y)) ^
    Kf(haslocation(?x)=haslocation(?y))
  </preconds>
  <effects>
    add(Kf, dead(?y));
  </effects>
</action>
```

With this information and a factual representation of the initial state the planner should return the following parametric plan. The plan involves checking Bill's location in the panel, driving to that location and killing Bill. The plan is not fully instantiated, as the actual location of Bill will only become known when the command is executed.

```
checklocation(bill)
drive(frolog,haslocation(bill))
kill(frolog,bill)
```

When the action `drive` is actually executed in the game, Bill's location can be obtained from the interaction KB because the action `checklocation(bill)` will already have been executed.

6.4 Concluding and linking the chapter

The discussion up to the point in this section might have seem too close to the implementation or linked to the particular purposes of a text-adventure game. We kept the discussion here as close to execution of physical acts as possible because we know how to formalize execution of physical acts in the framework of Frolog. However, the bridging of sensing acts can not only be used for execution. Sensing acts are acts that at bridging time are non-deterministic; that is, they generate information of the type *know whether* and *know value* which result in conditional plans and parametric plans. These plans which include uncertainty can be used to make clarification requests before execution. In our first case-study, the conditional plan obtained by the planner, which involves trying with both keys, can be used in order to make coherent next turn clarification requests such as: "Which key?". This is exactly the kind of CRs that we observe that people ask in these situations. Take for example the following fragment of the SCARE corpus. This is a fragment of the long example explained in Chapter 2. The DG just told the DF to put the rebreather into a particular cabinet, and they have just found the cabinet. Now, the DF wants to open the cabinet, and he knows that buttons in the room where cabinets are located open the cabinets. In this room there are two buttons. The left button is the closest to the target cabinet, and then this dialogue fragment happens:

DF(46): so how do I open it?
DF(47): one of the buttons?
DG(48): yeah its the left one
DF(49): alright makes sense [presses the left button and the cabinet opens]

The DF first makes explicit the precondition for which he cannot find a non-ambiguous plan with "so how do I open it" but he is not happy with just that. In turn (47) the DF makes explicit the fact that he knows that pressing one of the buttons in the room will probably achieve the desired effect of opening the cabinet. When the DG answers positively indicating that it's the left button in turn (48) the DF confirms its suspicion that the closest button is the one that will do the trick. The CR (47) can be generated from the conditional plans that involves pressing both buttons in order to find out which one opens the door. The DF does not need to go down to execution in order to acquire the information that he needs, as the DG has this information. In this exchange the DF need to reason on sensing actions (such as pressing buttons in the SCARE setup) and their effect on knowledge in order to generate the CR (47). Such sub-dialogues are commonplace in situated conversation such as the SCARE corpus.

Chapter 7

Conclusions and directions

7.1 Conclusions

The main goal of this thesis is to argue for a change of perspective in the study of conversational implicatures. Our proposal can be trivially stated: *Let's study conversational implicatures (CIs) in conversation.* But it is not so easy to put into practice. If I had to summarize the controversies and ideas that this perspective triggered during the writing, I would do it as follows (in a dialogue between the optimist ☺ and the pesimist ☹, both interested in CIs):

☺*(1): let's study CIs in conversation!*
☹*(2): how?*
☺*(3): observing them in a corpus of conversations*
☺*(4): and synthesizing them in a dialogue system*
☹*(5): by definition CIs are not said*
☹*(6): they are not explicit*
☹*(7): so you will not see them in a corpus, duuuh!!*
☺*(8): CIs are negotiable and clarifications are used for negotiation in conversation*
☺*(9): we could look for CIs in the clarifications of a conversation*
☺*(10): if the hearer asks about a CI, it can't be coincidence, we know he inferred it!*
☺*(11): and if he clarifies things that our system does not infer, we can improve it!*
☹*(12): wow wow wow how does your dialogue system infer CIs?*
☺*(13): well, you know, CIs are abductive inferences, inferences to the best explanation*
☹*(14): you are a computer scientist, you really expect abduction to be computational?!*
☺*(15): probably full abduction is not necessary*
☺*(16): we can use AI planning as a restricted kind of abduction*

In the rest of this Section we will expand and discuss the points made in this dialogue.

7.1.1 Study implicatures in conversation

The CI literature is based almost entirely on evidence obtained using introspective methods such as the *inference method* and the *verification method*. As discussed in Chapter 1, these two methods have

been shown to exhibit extremely different results over the same examples. The difference between these two methods is that the inference method puts the implicature under discussion while the verification method doesn't. As a result, the inference method fails to tell us anything about how utterances should be interpreted when the alleged implicature is *not* at stake, and the verification method fails in the opposite direction: it fails to tell us anything when the issue *is* at stake. What these results show is that *whether the CI is at stake or not* is crucial for the inference of the CI. Therefore, a lot is lost if the inference of CIs is studied in a contextualized fashion. CI have to be studied in their natural environment: conversation. This subsection expands on turns (1) to (4) which propose the task of defining "How to study implicatures *in conversation*".

The line of attack proposed in turn (3) is straightforward empirical analysis as is nowadays routinely done in the area of corpus linguistics. We applied this line of attack in Chapter 2 by directly exploring a corpus of human-human dialogues. The analysis performed in Chapter 2 allowed us to observe the emergent structure of dialogue in retrospective, and to gain insight into the relation between conversational implicatures and clarification requests in dialogue. However, the structure of a dialogue (which has already finished) is a trace of the opportunities *taken*, not of the opportunities *considered*. The clarification requests made in the dialogue make explicit those implicatures that *were inferred* and that one of the dialogue participants *decided to make explicit*. In a finished dialogue, there is no trace of those implicatures that a participant (either the speaker or the hearer) entertained but didn't make explicit. When the DG gives an instruction to the DF, the DG probably has in mind a particular way of carrying out the instruction, that is he has in mind implicatures of the instruction. Depending on how well coordinated the DG and the DF are, the DG will have higher or lower expectations that the DF will internally bridge all these implicatures exactly as the DG intended. If the DF decides to externally bridge any of them (that is, to clarify it) the DG will have to then support it (if he wants it executed) or deny it (if he didn't mean it). Even if the DG decides to support it the DF might disagree or propose to do it in a different way. Even in a constrained setup such as SCARE such interactions appear; in a less structured setup it is not difficult to imagine that the addressee of the project can alter it into a quite different project.

If conversation is analyzed only in retrospective, as done by empirical methods, the dynamics of the mechanism underlying conversation can only be guessed. This is why we consider the line of attack of analysis by synthesis (proposed in turn (4)) as a crucial complement to direct empirical analysis. In empirical analysis, the conversational mechanism is analyzed *after* the conversation is finished, in analysis by synthesis the mechanism is analyzed *while* the conversation is being produced. In Chapter 6 we implement the synthesis of CIs. Our implementation makes explicit the role that mutual information plays in the implemented conversational mechanism: if the CI can be inferred using practical reasoning on the mutual information then bridge it internally, if not, ask the DG to bridge it externally. This mimics the typical reaction that we observed in the human-human corpus.

7.1.2 Observing interactive implicatures: CRs $\rightsquigarrow$ CIs

This subsection expands on turns (5) to (10) which elaborate on the obstacles of *Observing interactive implicatures*.

Probably, the most obvious obstacle to observing CIs in conversation is that CIs are not explicit by definition (as noted in turn (6)). CIs are simply not in the corpus because they are not said. That

is probably a major reason why there are no corpus studies of CIs and the literature is based almost entirely on evidence obtained using introspective methods.

A solution to this problem was found by making use of one of the clearer distinguishing properties of CIs, namely that CIs are *negotiable* (turn (8) of our concluding dialogue). The hearer cannot be 100% sure that the speaker meant a CI, and the speaker is aware of this, so both of them can further talk about the CI, that is they can negotiate whether they will add it to their mutual knowledge or not. Conversation provides an intrinsic mechanism for carrying out negotiations of meaning, namely clarifications. In dialogue, clarification requests (CRs) provide good evidence of the implicatures that have been made because they make implicatures explicit. Therefore, what the CIs of an utterance are can be determined by exploring its CRs in corpora (turn (9) and (10)). This approach to the study of CIs was applied in Chapter 2; we called it CRs $\rightsquigarrow$ CIs in this thesis.

The second obstacle that we found was the rumor that so called pragmatic CRs (CRs that make CIs explicit) are rare in dialogue. This rumor, if true, would make the dialogue system community extremely happy because it would mean that they can treat CRs without heavy plan-recognition machinery. However, as we showed using the SCARE corpus (where roughly half of the CRs are pragmatic), when the hearer needs to figure out precisely the CIs of an instruction because they are necessary for the task at hand, he will ask about the CIs; that is, he will make pragmatic CRs. Therefore, the number of pragmatic CRs depends on the characteristics of the dialogue task. In Chapter 2 we discussed characteristics that increase the number of pragmatic CRs.

The third obstacle was that fact that the definition of CR turned out to be a hard one to pin down. Our approach to this problem was to use Herbert Clark's action ladder of communication. As a result, given an utterance, the following turn is its CR if its not its evidence of up-take. We do not claim here that this is the "right" definition of CR. However, it is good enough for our purposes. We need a definition of CRs that is sufficiently broad so it does not rule out the CRs that we are more interested in, that is, CRs in the pragmatic level. Using this definition we defined a "quasi-systematic' way of identifying CIs in corpora, something that the CI community desperately needs.

The main conclusion of this chapter is that the implicatures that are finally part of the discourse record are the exclusive responsibility of neither the speaker not the hearer. Both are involved in the process of deciding which CIs will be finally added and they do this through the interaction. Through this interaction the structure of the implicatures that both speaker and hearer are committed to *emerges*.

In the SCARE corpus, most of the CIs that we found seem to fall into the Grice category of **relevance implicatures**; 84% of them can be classified as either implicated premises, implicated conclusions or explicatures. It would be interesting to look for corpora in which other kinds of implicatures, such as **quantity implicatures**, are made explicit in CRs.

7.1.3 Synthesizing interactive implicatures: CIs $\rightsquigarrow$ CRs

This subsection expands on turns (11) to (13) which hint at the possibility of *Synthesizing interactive implicatures*. Levinson already highlighted the importance of analysis by synthesis for the study of face-to-face interaction in his [1983] book. He argues that such an analysis will make clear the minimal properties required for interaction; without these properties it wouldn't be possible to sustain any kind of systematic interleaving of actions by more than one party.

In the analysis by synthesis done in this thesis, two properties have turned out to be crucial: (1) an explicit representation of the mutual beliefs about the interaction domain, (2) and practical reasoning capabilities for using these mutual beliefs.[1] Chapter 4 shows that both of these ingredients are necessary for inferring the CIs that are made explicit in the CRs found in the SCARE corpus: In the framework developed in Chapter 4 (1) corresponds to the interaction model, which is implemented inside a conversational agent in Chapter 5 and (2) corresponds to the planning inference task , which is implemented inside a conversational agent in Chapter 6.

The CIs that are inferred using ingredients (1) and (2) can be either made explicit in a CR, that is **externally bridged** (for example, in Grice's garage example the guy standing by the car can say *and it's open?*) or silently accepted, that is **internally bridged** (for example, the guy just assumes that if the passerby knew that the garage is closed he would have said so and so assumes its probably open). The internal process of bridging is what in the literature has been called accommodation [Lewis, 1979] or bridging [Clark, 1975]. The external processes of bridging constitutes a large part of what we call conversation. In the system implemented in Chapter 6, the internal and external bridges are constructed using the *same* mechanism: performing practical inferences on the mutual beliefs. This approach to the traditional concepts of accommodation and bridging shed light on the current and traditional debates in these areas: such as presupposition failure and the principle of global accommodation (PGA).

If we take a common ground approach to the presupposition failure problem, the formulation of the problem goes along the following lines: what happens if a presupposition is false with respect to the (assumed) mutual beliefs? The crucial difference between this approach to the problem and the traditional one is that the information in the (assumed) mutual beliefs might be *incomplete* and *incorrect*. Since conversational partners are aware of this they might rather question a false presupposition instead of rejecting it.

In accommodation theory the PGA is assumed to hold but it is not known why it should hold. However, if we move to a bridging view of the phenomena we can argue that the addressee will accommodate the presupposition not into the global context but into the part of the common ground in which it can be bridged to, and if it cannot be bridged then the addressee will ask. Overhearers (who have less common ground with the speaker) seem to be much better candidates for the PGA principle (see Section 4.3.2 for an example). Understanding how overhearers deal with accommodation is beyond the scope of this thesis; we take an addressee perspective on interpretation. We can speculate that overhearers will add presuppositions to the global context simply because they have reached the global context without being able to build any bridge and must leave the information "at hand" just in case they get the chance to ask about it latter. But this is only a speculation. A theory of interpretation should make explicit whether it takes an addressee or an overhearer perspective. If accommodation theory stopped analyzing language as an overhearer and tried to model the accommodation that happens routinely in everyday dialogue where interlocutors are using language for actually doing things, the theory might develop in interesting new directions.

[1] These two ingredients closely correspond to three of the ingredients predicted by Levinson [1983, p. 44]: shared knowledge of a domain and its update, and means-ends reasoning for interpreting intentions. The other three properties predicted by Levinson are related to generation, such as means-ends reasoning for generating intentions and formation of overall goals.

7.1.4 Inferential concerns and insights

This subsection expands on turns (14) to (16) which highlight the computational cost of using abduction and to the idea proposed in this thesis of using planning (and in particular, non-classical extensions of planning) as a constrained kind of abduction.

The BDI framework for dialogue management is still the most evolved theoretical framework but has been largely replaced in practical applications (because of its computational complexity) by shallower methods [Jurafsky, 2004]. BDI approach to the inference of conversational implicatures is based on the plan recognition inference task. This approach implicitly adopts the view of conversation that Herbert Clark calls the *goals-and-plans view* that is the whole conversation is viewed a plan.

The approach adopted in this thesis is different from BDI in two essential ways. First, we use planning, and not plan-recognition, for interpretation. Second, plans are used to bridge the gap between an utterance and its context. That is, each utterance (and not the whole dialogue) corresponds to a plan. Our approach then can be seen as adopting the the view of conversation that Herbert Clark calls the *opportunistic view*.

The key insight behind our approach is that we view planning as a restricted kind of abduction for interpretation. It is a known fact that planning is a kind of abduction [Eshghi, 1988] — the actions are the theory, the goal is the observation and the initial state is the set of abducibles — in which all the assumptions have to be eventually grounded in the initial state. It can be argued that, in virtue of the fact that it uses planning, our framework provides a model of textual coherence, forcing the necessary assumptions to be linked to the previous context instead of allowing for the assumption of arbitrary propositions. The two paradigmatic approaches to the use of abduction for interpretation (Hobbs's and Charniak's) presented in Chapter 3 allow for the assumption of arbitrary propositions and lack a model of coherence.

The problem with classical planning is that the model of textual coherence that it provides is too constrained. Classical planning forces all the observations to be explained to be eventually connected with the initial state through a plan. Hence, the initial state must contain all the information that is relevant for the interpretation of the observation; that is, it has to contain complete information.

One of the main claims of this thesis is that planning extended with incomplete knowledge and sensing is a good comprise, allowing missing information to get into the state in a constrained fashion (namely through sensing actions). To put it in another way: it allows us to go one step higher in the abduction hierarchy while maintaining low computational complexity.

We note in passing that the planning extended with sensing representation allows for the explicit representation of and the reasoning on the **Competence Assumption**. The Competence Assumption is the assumption that an agent *knows whether* some proposition is the case or not. This assumption is necessary for strengthening weak implicatures. See [Geurts, in press] for a thorough discussion of weak vs strong implicatures and the role of the Competence Assumption.

7.2 Directions

In this section I summarize the theoretical links between the framework that has been developed in this thesis and the theories discussed along the thesis. These links suggest (long term) directions that might be taken to tackle phenomena not so far incorporated in the framework presented here, but

also directions that might be taken to resolve problems which arise in previous work on implicatures and accommodation. The last subsection summarizes the (short term) future work that we think is needed in order to validate the empirical and methodological claims made by this thesis on a larger scale.

7.2.1 For dialogue systems: tacit acts

In Chapter 6 of this thesis we implemented the inference framework we proposed in Chapter 4 in a dialogue system situated in a text adventure game (the architecture of the general system is introduced in Chapter 5). The system interprets instructions that the user gives and executes them in a simulated world. If an instruction cannot be directly executed because it has requirements that are not fulfilled by the current state of the world the system starts an internal process of bridging using practical inference and tries to infer the implicated premises of the instruction with respect to the current context as tacit dialogue acts. If the internal process is successful then the system executes the instruction and updates the context with it and its implicatures. If the internal bridging process fails the system starts an external bridging process verbalizing the obstacles and prompting in this way a repair sub-dialogue.

Other researchers in the dialogue system community have tried to implement systems that have internal bridging (i.e., accommodation) abilities using tacit acts. Kreutel and Matheson formalize the treatment of tacit dialogue acts in the **Information State Update** framework [Larsson and Traum, 2000]. According to Kreutel and Matheson, context-dependent interpretation is ruled by the following principle:

> ***Context Accommodation (CA)**: For any move* m *that occurs in a given scenario* sc_i*: if assignment of a context-dependent interpretation to* m *in* sc_i *fails, try to accommodate* sc_i *to a new context* sc_{i+1} *in an appropriate way by assuming implicit dialogue acts performed in* m*, and start interpretation of* m *again in* sc_{i+1}*. [Kreutel and Matheson, 2003, p. 185]*

The authors concentrate on the treatment of implicated acceptance acts but suggest that the CA principle can be seen as a general means of context-dependent interpretation. In [Kreutel and Matheson, 2003] the problem of *how* such implicit dialogue acts are to be inferred is not addressed.

Thomason *et al.* formalize the treatment of tacit dialogue acts in the **Contribution Tracking** framework [DeVault, 2008] arguing that pragmatic reasoning is continuous with common-sense reasoning about collaborative activities and is ruled by the following principle:

> ***Enlightened Update (EU)**: C is a condition that can be manipulated by an audience H. An agent S is observed by H to be doing A while C is mutually known to be false. H then acts [tacitly] to make C true, and S expects H to so act. [Thomason et al., 2006, p.36]*

In this way they argue that, in order to make pragmatic inferences such as CIs, there is no need to distinguish cases in which A is a physical act and cases in which A is a speech act. Thomason *et al.*'s EU principle can be seen a refinement of Kreutel and Matheson's CA principle. The EU principle specifies when the assignment of a context-dependent interpretation fails: when a condition C of the action A is mutually known to be false. Moreover, the EU framework stipulates that the tacit acts

will be inferred using abduction on the common ground and the specification of action moves. Since the framework uses abduction, it suffers from the ambiguity resulting of the unconstrained production of potential tacit acts.

Devault and Stone [2009] approach this problem using two strategies. On the one hand, their implemented system handcrafts domain dependent constraints. On the other hand, the system learns such constraints through interactions with a human user. The system accumulates training examples which track the fates of alternative interpretations (including its tacit acts) across dialogue. The fates are frequently determined by subsequent clarificatory episodes initiated by the system itself (that is, if the system does not know which tacit act — i.e., which CI — the user implicated, it will ask the user).

Devault *et al.* 's approach has only been used up to now for implementing a quite small number of actions (they implemented a system that can collaboratively refer to objects using a small number of properties). A second problem that Devault *et al.* 's approach to CIs will have to face when modeling bigger domains is computational complexity and an even more severe proliferation of ambiguity due to the inference of the tacit acts. For facing such challenge in an effective way we believe that the combination of planning as a constrained kind of abduction and the learning of constraints from interaction are a promising combination.

And what about generation?

In chapter 1 we said that the picture of granularity in conversation presented there open up two big questions. The first can be described as the generation problem and the second as the interpretation problem: 1) How do speakers choose the bit of information that they will make explicit in a coarse-grained level (that is, why did my mom choose to say *Buy food for Tiffy* and not some other information in the segment)? 2) How do addressees reconstruct, from the coarse-grained level that they observe, the fine-grained level that the speaker means? In this thesis we focused on question (2). However, the lessons learned while exploring question (2) can be used to shed light on question (1), since (at some level of abstraction) generation and interpretation are two sides of the same coin. Generation can be thought of the process of producing a surface form from which the addressee can reconstruct the intended message. What's more, if a message is generated with particular CIs in mind, those CIs should be kept in mind since it is to be expected that the addressee might have doubts about them in following turns. Such an approach could be evaluated, for instance in one of the current Challenges of the Natural Language Generation community, namely the GIVE challenge [Byron *et al.*, 2009].

7.2.2 For inference: non-classical practical reasoning

Frolog is a proof of concept that practical reasoning tasks that have been and are being intensively investigated in the area of Artificial Intelligence are usable for inferring conversational implicatures in a contextualized fashion. The off-the-shelf reasoning tasks that Frolog integrates are *classical planning* as implemented by the planner BLACKBOX [Kautz and Selman, 1999], and *planning with knowledge and sensing* as implemented by the planner PKS [Petrick, 2006]. These are generic reasoning tools that were not implemented with the inference of conversational implicatures in mind and we have proved that they can be used effectively for this task.

The use of such generic tools has several advantages. To start with, the area of Planning in

Artificial Intelligence is a rapidly growing area where powerful optimization to the reasoning algorithms have been implemented. The area of conversational implicatures can directly benefit from these optimizations to scale up the phenomena that can be implemented inside a real-time dialogue system. Furthermore, the area of planning has been exploring the integration of planning reasoning tasks with the treatment of stochastic based models of uncertainty and with artificial intelligence learning techniques. The International Planning Competition has recently included a Learning Track and a Uncertainty track apart from its traditional classical deterministic planning tracks. The area of conversational implicatures can probably benefit from the advances to come in these areas, which might implement the "Fuzzy practical reasoner" [Kenny, 1966] foreseen by Brown and Levinson [1978, p. 65] as a basic component for computing conversational implicatures.

However, as happens with any tool that is being used in an unusual way, there are disadvantages as well as advantages. Let us start with a very straightforward disadvantage that is not clear to us how to overcome.[2] When a planner cannot solve a planning problem it says "there is no plan" and gives no further explanation. People are much more informative when they do not find a link to the previous context as illustrated by one of the examples discussed in Chapter 3 and reproduced here:

DG(1): we have to put it in cabinet nine .

The instruction DG(1) has the precondition *the reference "cabinet nine" is resolved*. However, this precondition cannot be achieved by any action because the DF doesn't know the numbers of the cabinets. Hence, a planner can find no plan for this planning problem. Our planning framework then knows that the goal *the reference "cabinet nine" is resolved* cannot be achieved and predicts a CR such as the following:

DF(2): what do you mean by 'cabinet nine'?

However the DF does not utter this CR but a more informative one, probably because he is able to infer *where* in the planning space it is not possible to link the plan with the initial state. That is, the DF finds an *explanation* of why the bridging failed (the cabinets are not numbered) and utters it as follows:

DF(2): yeah, they're not numbered

The question for the planning community would be: can planners explain the reasons why they did not find a plan?

A second wish in our wish-list would be to have planners that can output more than one plan. This is an ability that no current off-the-shelf planner offers to the best of our knowledge; probably because it's an ability not required by the International Planning Competition. But this will be required in order to implement the multiple plans prediction of CRs designed in Chapter 4. According to people in the planning community, while it is easy to let FF [Hoffmann and Nebel, 2001] or PKS [Petrick and Bacchus, 2004] planners go on with the search after finding the first plan, it is probably not easy to make it aware of any dependencies the returned plans should have.[3]

[2] We have recently learned (p.c. with Jörg Hoffman) that the planning community has been working on a related problem called landmarks [Hoffmann *et al.*, 2004], this seems a promising direction that we plan to address in further work.

[3] p.c. with Ron Petrick and Jörg Hoffman.

These are two topics the planning community have not looked into but seem generic enough to interest them. In any case, the conversational implicature community and the planning community have a lot to gain if they start to talk.

7.2.3 For sociology: societal grounding

In the preliminary study of the SCARE corpus that we did, the corpus presents more CRs in general and in particular more CRs in level 4 (pragmatic CRs) that those reported in previous studies [Rodríguez and Schlangen, 2004; Rieser and Moore, 2005; Ginzburg, in press]. In Chapter 2 we argued that this is explainable in terms of the task characteristics. However, social pressures, such as those associated to politeness strategies [Brown and Levinson, 1978] also play their role.

The participants of the SCARE experiment were punished if they performed steps of the task that they were not supposed to (see the instructions in Appendix A). This punishment might take precedence over the dis-preference for CRs that is universal in dialogue due to politeness. CRs are perceived as a form of disagreement which is universally dis-preferred according to politeness theory. Moreover, the pairs of participants selected were friends so the level of intimacy among them was high, lowering the need of politeness strategies; a behavior that is also predicted by politeness theory.

The study of the interaction between politeness constraints and clarification strategies seems to be an important one to keep in mind, and we plan to address it in future work. In this thesis, we have found that blurring the border with sociolinguistics is extremely rewarding when trying to find explanations for the phenomena we found in the data. In particular, we have found inspiration in the works on politeness theory [Brown and Levinson, 1978] and on repair [Schegloff, 1987a; Schegloff, 1987b].

The effect that sociological factors have on the use of language is frequently overlooked in the area of pragmatics; though there are some notable exceptions. DeVault et al [2006] argues that an essential part of using language meaningfully (a concern that is certainly pragmatic) is working to keep your meanings aligned with what others mean in the society. DeVault et al [2006] call this process **Societal Grounding** while Larsson [2007] calls it **Semantic Coordination**. Both of them argue that such processes explain how language coordination occurs in the adaptation of communicative resources (which include linguistic resources) to specific activities or situations. Larsson argues that this might be the same kind of process that underlies **language change** over long time periods (as studied in historical linguistics) but at a micro-scale. In this view, meaning is negotiated, extended, modified both in concrete situations and historically. This means is that we cannot hope to write down today and for good all the pragmatic rules that are out there. Pragmatics will have to study the processes: how pragmatics rules are acquired, how they evolve and how they are exploited.

7.2.4 For the philosophical origins of implicatures

In Chapter 4 we formalized Grice's famous example of relevance implicature (i.e. the garage example) using the framework developed in the same chapter. Let me briefly summarize here the procedure that we used and which could, be applied to other kinds of conversational implicatures. We reproduce the example here once again Grice's garage example:

(19) *A: I am out of petrol.*
B: There is a garage around the corner.
B conversationally implicates: the garage is open and has petrol to sell.
[Grice, 1975, p.311]

We argued that the *bald on record* contributions — that is *assertional implicatures* in Thomason's terms [Thomason, 1990] and *explicatures* in relevance theoretic terms [Wilson and Sperber, 2004] — behind this contribution were the following:

(20) *A: I'm out of petrol))))) A: Tell me where to get petrol*

(21) *B: There is a garage around the corner))))) B: Get petrol in a garage around the corner*

The calculation of such explicatures is beyond the abilities of Frolog. Frolog can infer only missing parameters in its current implementation and not politeness strategies. Frolog could be extended in order to integrate some more elaborate explicature inference using the methods introduced in Section 1.2.4 (such as the *cue-based model*) for speech act identification.

Assuming that the explicatures where calculated, our framework calculates the implicated premises as illustrated in Figure 7.1 reproduced below.

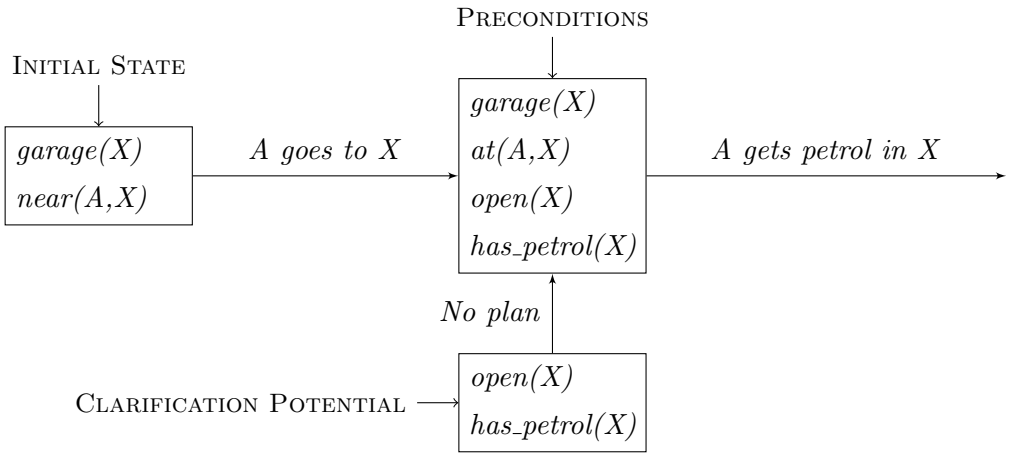


Figure 7.1: Inferring the implicated premises of Grice's garage example

Since there is no plan that explains the preconditions *open(X)* and *has_petrol(X)* then our framework predicts that the following two are potential CRs that can be uttered next:

(22) a. *A: and you think they have petrol?*
b. *A: and you think I am at the garage?*

Since a plan can be found that explains the preconditions *garage(X)* and *at(A,X)* then the framework does *not* predict that there will be CRs like the following although these are also preconditions of "Getting petrol in X":

(23) a. *A: and you think it's open?*
b. *A: and the garage is a garage?*

So if we came back to Grice's terminology, we can say that the fact that the garage is open and has petrol to sell has been implied by B, and the evidence for this is that these facts are predicted as readily negotiable in our framework.

It has been argued[4] that a person, that is not A and B and hears this exchange, does not need to infer the CIs discussed above in order to understand the coherence of this exchange. Let's call this person an **overhearer**. And it is a fact that overhearer might not need to go all the way down to the task level of granularity (see distinction between addressees and overhearers in [Clark, 1996, p. 151]). The level of granularity that is enough for an overhearer to say that he has understood the exchange need not coincide with the level that is required by the addressee and depends on what the interest of the overhearer in the conversation is, and what the information that he is expecting to take out of the exchange is. However, addressees *do* have to go all the way down to the task granularity (that is, to level 4 in Clark's terms, see Section 2.1.2). In the Grice example, it is relevant for A's goals that the garage is open so A will draw this inference from what B said.

The important element for the inference of the CI is that the CI has to be particularly important for the dialogue participant's goals. For instance, suppose that the garage exchange happens at 4am, it wouldn't be uncooperative if the exchange continues:

(24) *A: and you think it's open?*

To know whether the garage is open or not is relevant for A's goals and it is not so probable if the exchange happens at 4am because at that time the garage might be closed. In task-oriented dialogue, in very cooperative setups, many CIs are asked to be made explicit because the task at hand requires a great deal of precision (and not because the task at hand is antagonistic as in criminal trials [Levinson, 1983; Asher and Lascarides, 2008]), and because the CI is not as probable in the current context.

Since CI are said to arise from the assumption of underlying cooperation in dialogue it has been argued [Levinson, 1983] that in antagonistic dialogue the CIs normally associated with what is said would not routinely go through, the hearer will not calculate them. A typical example of this argument is set in a cross-examination in a court of law and the exchange goes as follows:

(25) *C: On many occasions?*
W: Not many
C: Some?
W: Yes, a few [Levinson, 1983, p.121]

The argument says that in cooperative dialogue the answer *Not many* would implicate *on some occasions* and then the following question by the counselor would not be necessary. The argument continues saying that here C's question is necessary since the CI does not arise given the antagonistic nature of the dialogue. However, if this argument is correct then it is a *surprising coincidence* that, at that point, C decides to question exactly the content of the CI. A more appealing explanation, I think, would be that C does infer the CI (even in this non-cooperative dialogue) and sees this as a perfect opportunity to make a point that is crucial to this legal process, namely that C did something *at least in some occasions*. In this view, cooperativity is not a prerequisite for calculability and then CIs do arise even in non-cooperative dialogue.

[4] p.c. with Manfred Pinkal.

7.3 Short-term future work

After finishing with this thesis we feel that we have only scratched the surface of two lines of work for studying conversational implicatures: empirical analysis and analysis by synthesis. With our work we have just provided proofs of concept that show that these two lines of work are indeed promising.

The empirical results we present here are suggestive but preliminary; we are currently in the process of evaluating their reliability measuring inter-annotator agreement. We are considering the GIVE challenge [Byron *et al.*, 2009] as a possible setting for evaluating our work (our framework could predict potential clarification requests from the users).

The analysis by synthesis that we performed already show the kind of detailed analysis of contextual information resources required for inferring conversational implicatures. However, the approach should be evaluated on larger conversational scenarios in order to evaluate its scalability; which is though promising as a result of the locality of the required inferences. The other straightforward line of work is further exploiting the abilities of non-traditional off-the-shelf planners such as those based on probabilistic grounds.

There is lot to do yet, but we believe that the interplay between conversational implicatures and clarification mechanisms will play a crucial role in future theories of communication.

Appendix A

Empirical methodology on the corpus

A.1 Decision procedure used in the annotation

The empirical results discussed in Chapter 2 and Chapter 4 were obtained following the methodology described here. The corpus SCARE [Stoia *et al.*, 2008] was selected according to the desired characteristics identified and discussed in Chapter 2. Our hypothesis was that such characteristics, which were exhibited in the SCARE corpus maximize the quantity of clarification requests (CRs) in level 4. Once the corpus was chosen, its fifteen dialogues were informally analyzed watching the associated videos while reading the transcripts to do a first confirmation of our hypothesis before starting with the systematic annotation procedure. This first informal analysis resulted in promising results, with many clear instances of CRs in level 4 identified, so we decided to proceed with the systematic annotation. For this we randomly selected one of the dialogues (without including in the set of candidates dialogue 1 which has almost no feedback from the DF because he thought that he was not supposed to speak). The dialogue randomly selected is dialogue number 3 in the SCARE corpus; its transcript contains 449 turns and its video lasts 9 minutes and 12 seconds.

The decision graph used in our "quasi-systematic" annotation is depicted in Figure A.1. We call our procedure "quasi-systematic" because, while its tasks (depicted in rectangles) are readily automatizables, its decision points are not and require subjective human judgments. Decision points D1 and D2 decide whether the turn is a CR or not; new tasks and digressions from the current task answer "no" to both decision points and just stack their evidence of proposal in T3. If the turn is a CR of a proposal X, T4 unstacks all proposals over X as a result of applying the downward evidence property of conversations (discussed in Section 2.1.2). Intuitively, the turn is taken as an implicit uptake in level 4 of all the proposals over proposal X (which must be completed before X can be completed).[1] Decision points D3 to D6 decide whether the CRs belong to Clark [1996]'s levels 1 to 4 respectively using Rodríguez and Schlangen [2004].

[1] This intuition is in line with Geurts's preliminary analysis of non-declaratives: The speaker did not negotiate the proposals over X then we can assume that he did not have problems up-taking them [Geurts, in press].

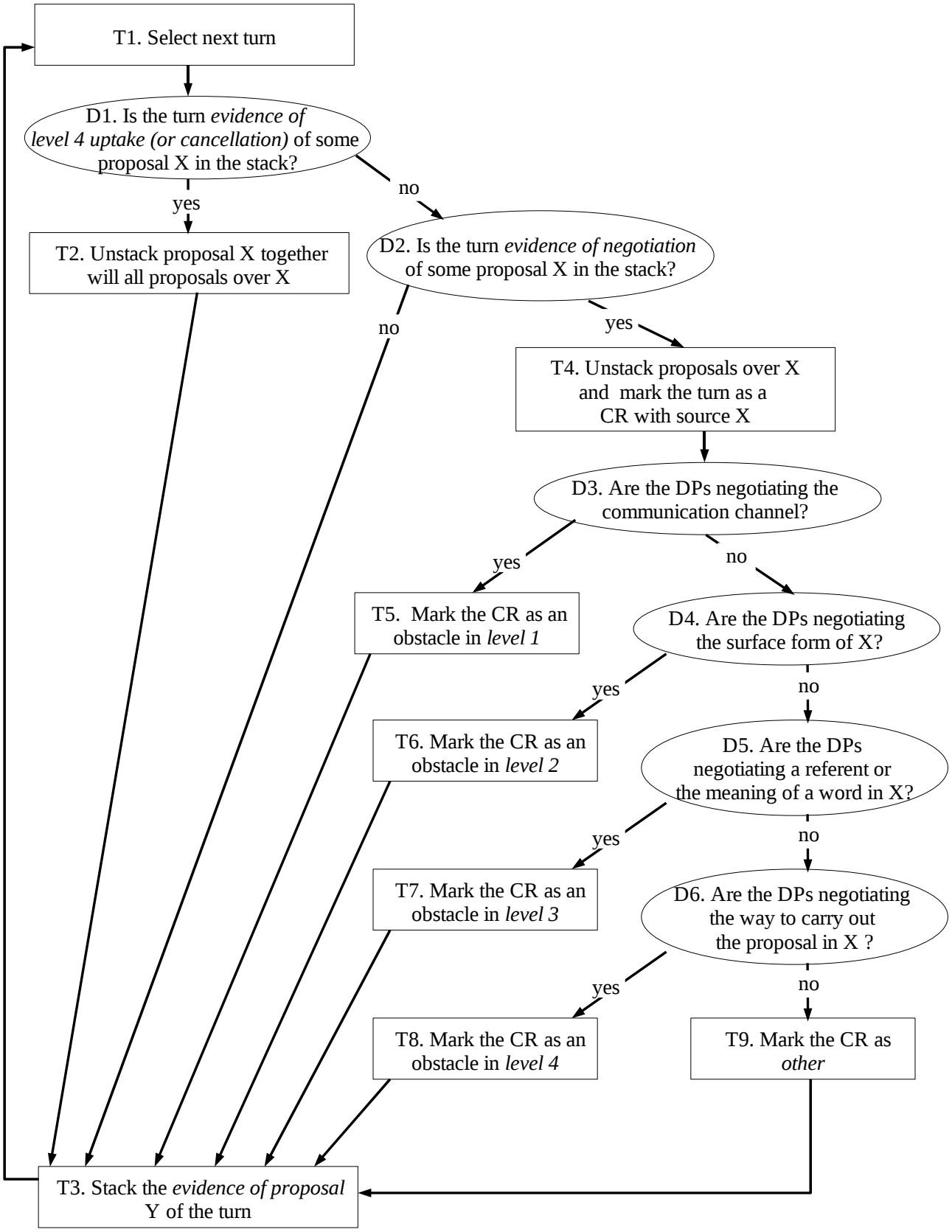


Figure A.1: Decision graph for annotating CRs in the SCARE corpus

A.2 Information available before the dialogues

In this section, we specify the information that was available to the DG and the DF before the SCARE experiment started (adapted from [Stoia, 2007]). These instructions are crucial for our study since they define the content of the information resources of the inference framework described in this thesis.

The following specification of the QUAKE controls, that is, the possible actions in the simulated world, were received by all participants.

1. Use the arrow keys for *movement*: ↑ (walk forward), ↓ (walk backward), → (turn right) and ← (turn left).

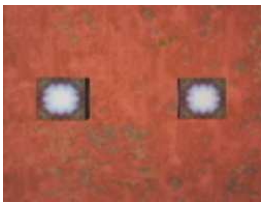
2. To *jump*: use Spacebar.

3. To *press a button*: Walk over the button. You will see it depress.

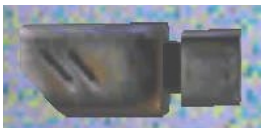
4. To *pick up an object*: Step onto the item then press Ctrl (Control key).

5. To *drop an object*: Hit TAB to see the list of items that you are currently carrying. Press the letter beside the item you wish to drop. Press TAB again to make the menu go away.

The participants also received the following pictures of possible objects in the simulated world so that they are able to recognize them.

Buttons	Cabinet
	

The following things were indicated as being objects that the DF can pick up and move:

Quad damage	Rebreather	Silencer
		

They also received the following warning: you will not be timed, but penalty points will be taken for pushing the wrong buttons or placing things in the wrong cabinets.

Apart from these instructions which were common to both dialogue participants, each of them received their own instructions. The instructions for the DF are described in Section A.2.1 and the instructions for the DF are described in Section A.2.2.

A.2.1 Instructions for the Direction Follower

Phase 1: Learning the controls First you will be put into a small map with no partner, to get accustomed to the QUAKE controls. Practice moving around using the arrow keys. Practice these actions:

1. Pick up the Rebreather or the Quad Damage.
2. Push the blue button to open the cabinet.
3. Drop the Quad Damage or the Rebreather inside the cabinet and close the door by pushing the button again.

Phase 2: Completing the task In this phase you will be put in a new location. Your partner will direct you in completing 5 tasks. He will see the same view that you are seeing, but you are the only one that can move around and act in the world.

Implications for the Potential Actions

In phase 1, when the DF is learning the controls, he learns that buttons can have the effect of opening closed cabinets and closing open cabinets. Such action is formalized as follows in PDDL [Gerevini and Long, 2005] and is included in the possible action database:

```
(:action press_button
   :parameters (?x ?y)
   :precondition
       (button ?x)
       (cabinet ?y)
       (opens ?x ?y)
   :effects
       (when (open ?y) (closed ?y))
       (when (closed ?y) (open ?y)))
```

Notice that this action operator has conditional effects in order to specify the action more succinctly. However, it is not mandatory for the action language to support conditional effects. This action could be specified with two actions in which the antecedent of the conditional effect is now a precondition.

In the following section we compare the content of the potential action database with the content of the world action database and its relation with one of the main sources of CRs in the SCARE dialogue, namely the mis-coordination of the potential actions and the world actions.

A.2.2 Instructions for the Direction Giver

Phase 1: Planning the task Your packet contains a *map* (reproduced in Figure A.2) of the QUAKE world with *5 objectives* that you have to direct your partner to perform. Read the instructions and take your time to plan the directions you want to give to your partner.

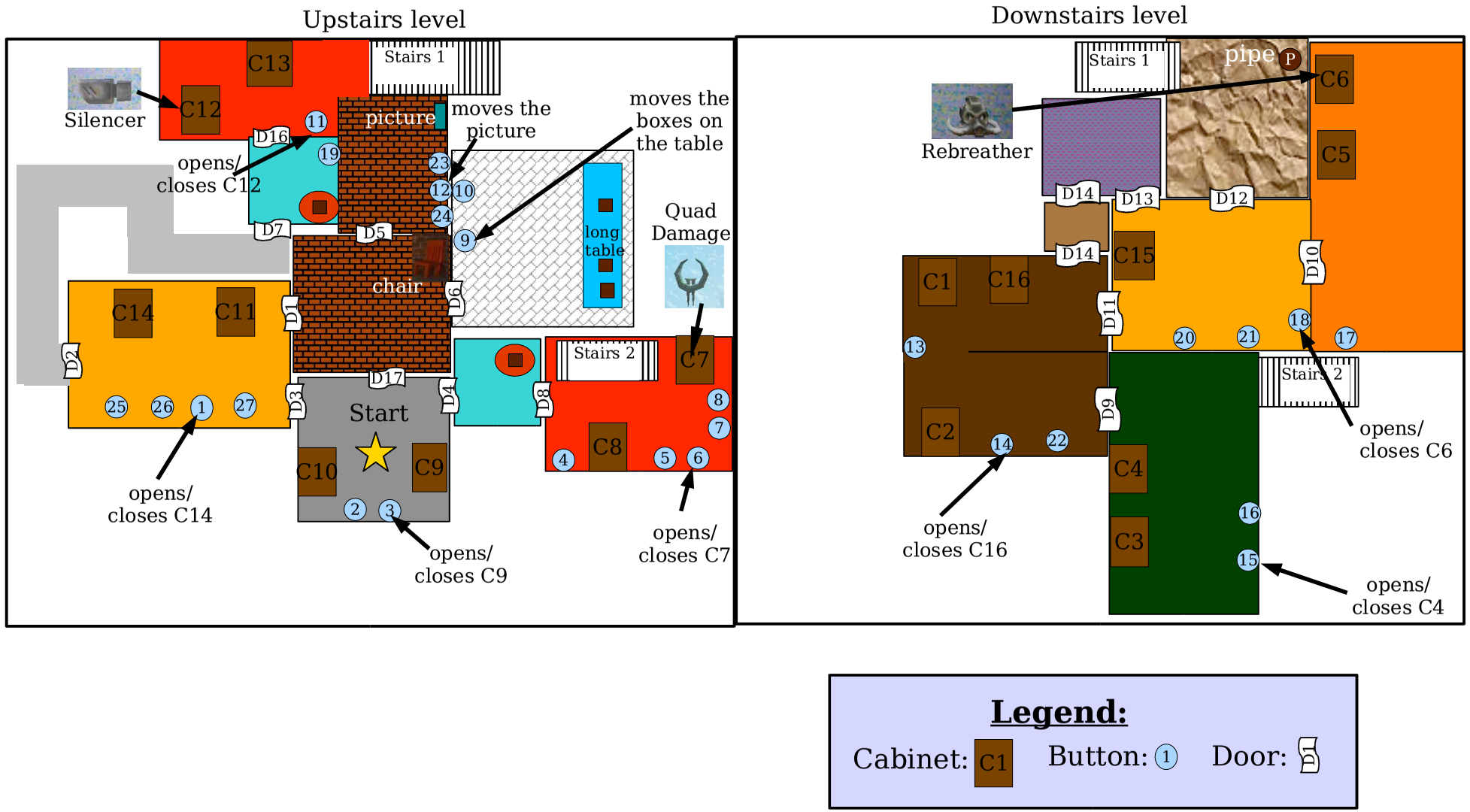


Figure A.2: Map received by the DG

Phase 2: Directing the follower In this phase your partner will be placed into the world in the start position. Your monitor will show his/her view of the world as he/she moves around. He/she has no knowledge of the tasks, and has not received a map. You have to direct him/her through speech in order to complete the tasks. The objective is to complete all 5 tasks, but the order does not matter.

The tasks are:

1. Move the picture to the other wall.
2. Move the boxes on the long table so that the final configuration matches the picture below.

Picture	Long table
	

3. Hide the Rebreather in cabinet 9. To *hide* an item you have to find it, pick it up, drop it in the cabinet and close the door.
4. Hide the Silencer in cabinet 4.
5. Hide the Quad Damage in cabinet 14.
6. At the end, return to the starting point.

Implications for the World Actions

The functions of the buttons that can move things can be represented in the following action schema. If the thing is in it's original location (its location when the game starts), we say that is thing is `not-moved`. If the thing is in the goal position then we say that the thing is `moved`. The specification of this action schema in PDDL is as follows.

```
(:action press_button_2
   :parameters (?x ?y)
   :precondition
       (button ?x)
       (thing ?y)
       (moves ?x ?y)
   :effects
       (when (moved ?y) (not-moved ?y))
       (when (not-moved ?y) (moved ?y)))
```

The world action database includes the PDDL action `press_button` as defined in Section A.2.1 and also the PDDL action `press_button_2` above. However, the potential actions only include `press_button` and not `press_button_2`. That is to say that the two action databases used are mis-coordinated which result in finding wrong plans using the potential actions, plans that cannot be execute din the SCARE world, as discussed in Chapter 4.

Implications for the World Model

The world model is a relational model that represents the information provided by the map, including the functions of the buttons and the contents of the cabinets. The Figure A.3 shows a fragment of the model of the world according to the specification of the SCARE experiment.

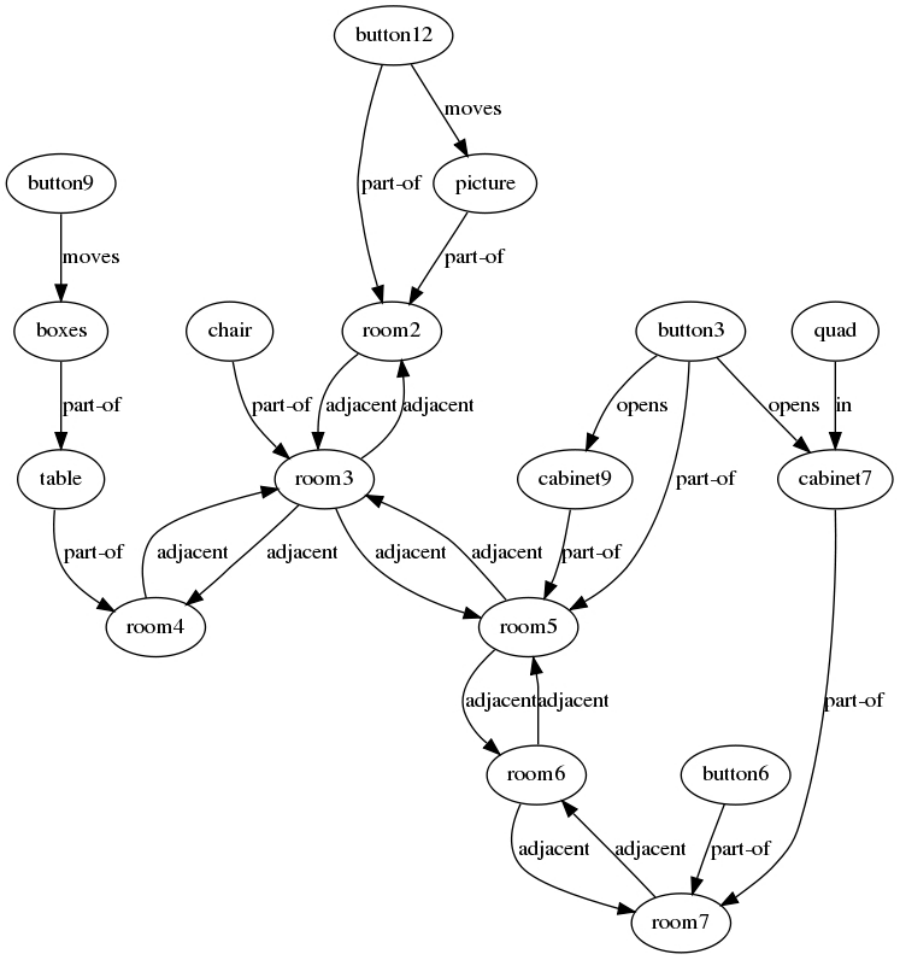


Figure A.3: Fragment of the SCARE world model

Appendix B
Planning Domain Definition Language

The Planning Domain Definition Language (PDDL) is intended to express the "physics" of a domain, that is, what predicates there are, what actions are possible, and what the effects of actions are. Most planners require in addition some kind of "advice" or control information in order to guide the search they perform. However, PDDL provide no advice notation and its authors explicitly encourage planner developers to extend its basic notation in the way required by the particular planner implementation.

Even without these extensions, few planners will handle the entire PDDL. Hence, this language is factored into subsets of features, called *requirements*. Every domain defined using PDDL should specify which requirements it assumes. Any planner then can easily determine if it will be able to handle certain domain specification. The syntactic requirements supported by PDDL that are relevant for this thesis are: basic STRIPS-style; typing; conditional effects; universal and existential quantification; and domain axioms.

B.1 PDDL with STRIPS and typing

We start our discussion by restricting our attention to the simple PDDL subset that handles STRIPS planning tasks in a deterministic, fully-specified world. In other words, both the preconditions and effects of actions are conjunctions of literals. After covering the basics, we describe the PDDL subsets with added features which offer a more expressive action language.

In what follows, we will describe PDDL syntax while introducing its semantics intuitively through a worked out example. For a complete definition of its semantics see [Gerevini and Long, 2005]. The planning problem example we will use in this section is based on the FairyTaleCastle scenario provided with FrOz. In the initial state of this example, a door is locked and its key is on a table, and the goal is to arrive at a state where a given door is unlocked.

Before describing how PDDL specifies the planning domain and the planning problem, we will describe the type system supported by PDDL. Types should be defined as part of the planning domain. Typing is important when defining planning tasks because it reduces the search space for those planners that ground the specification before starting the search.

0) Types: PDDL allows for the definition of types that can then be used to type both objects in the planning problem and the arguments of predicates and actions.

In our example we can declare the types:

```
(:types
   key takeable lockable table
   player room container door top)
```

where `top` is a type that is applied to all the individuals in the planning problem in order to symplify the typing of paramenters.

In general, the type definition in PDDL is specified as:

$$(\texttt{:types <type-list>*}) \tag{B.1}$$

where `<type-list>` is a list of identifiers (type names).

The predicates involved in the planning task can be declared and the type of their parameters specified as shown in the following example:

```
(:predicates
   (haslocation x - top y - container)
   (locked x - lockable)
   (fitsin x - key y - lockable))
```

The predicate `(haslocation x y)` holds when `x` is located in the container `y`, `(locked x)` holds when the lockable object `x` is locked, and `(fitsin x y)` holds in the world when the key `x` opens the lockable object `y`.

In general, predicate definitions in PDDL are specified as:

$$(\texttt{:predicates } \{\texttt{<ident> } \{\texttt{<variable> - <type>}\}*\}*) \tag{B.2}$$

where `<ident>` is the predicate name followed by a list of elements formed by a variable name `<variable>` and its type `<type>`.

1) Initial State: Remember that the initial state is a description of the world state when the plan begins. It includes the objects available in the planning problem as well as a specification of which predicates are applicable to them, when the plan begins. Since types are simply a specific kind of predicate which have the particularity that cannot be affected by action effects, object types are also specified here.

Returning to our example, the object and initial state definitions will be as follows:

```
(:objects
   empfang - (either room container top)
   myself - (either player container top)
   door1 - (either door lockable top)
   key1 - (either key takeable top)
   table1 - (either table container top))

(:init
```

```
(haslocation myself empfang)
(haslocation key1 table1)
(haslocation door1 empfang)
(haslocation table1 empfang)
(locked door1)
(fitsin key1 door1))
```

In general, the object definition in PDDL is specified as:

$$(\texttt{:objects } \{\texttt{<ident> - (either } \{\texttt{<type>}\}*\texttt{)}\}*) \tag{B.3}$$

where `<ident>` is an identifier for the name of an object followed by a list of types to which it belongs. In this definition `either` is the PDDL reserved word which indicates that some object belongs to more than one type. And, in general the initial state definition in PDDL is specified as:

$$(\texttt{:initial } \{\texttt{(<ident>}\{\texttt{<object>}\}*\texttt{)}\}*) \tag{B.4}$$

where `<ident>` is the name of some predicate and `<object>` is the name of some of the defined objects.

Most planners that accept PDDL planning specifications assume that the initial state is closed (they implement a Closed World Assumption), i.e., any fact about the initial state not explicitly indicated is assumed to be false.

2) Goal: The goal is a description of the intended world state but unlike the initial state it is usually not a complete description of the world. Any state that satisfies the literals included in the goal is acceptable as a goal state.

Remember that in our example the goal is a world in which the door is unlocked. This can be expressed in PDDL in the following way:

```
(:goal
   (no-locked door))
```

In general the goal definition in PDDL is specified as:

$$(\texttt{:goal } \{\texttt{(<ident>}\{\texttt{<object>}\}*\texttt{)}\}*) \tag{B.5}$$

where `<ident>` is the name of some predicate and `<object>` is the name of some of the defined objects.

3) Domain: The domain includes a crucial element in planning: the *actions*. Actions are schemes whose parameters are instantiated with the objects defined for the planning problem. Each action specifies three elements:

- Action name and parameter list.
- Preconditions: a conjunction of literals that state which are the conditions that must be satisfied by the world in order to be able to execute the action.
- Effects: a conjunction of literals that describe how the world changes when the action is executed.

Two sample actions in PDDL are:

```
(:action take
   :parameters (?x - takeable ?y - container)
   :precondition
     (not(haslocation ?x myself))
     (haslocation ?x ?y)
   :effect
     (not(haslocation ?x ?y))
     (haslocation ?x myself))

(:action unlock
   :parameters (?x - lockable ?y - key)
   :precondition
     (locked ?x)
     (haslocation ?y myself)
     (fitsin ?y ?x)
   :effect
     (not(locked ?x)))
```

The first action allows the object `myself` to take an object that is takeable (`?x - takeable`), which `myself` is not holding already (`not(haslocation ?x myself)`) and that is located in some container `?y` (`(haslocation ?x ?y)`). The action has two effects: the object is relocated to `myself` (`(haslocation ?x myself)`) and hence, it no longer located where it used to be `(not(haslocation ?x ?y))`. The second action can be interpreted similarly.

B.2 Handling expressive PDDL

Until now, our discussion has been restricted to the problem of planning with the STRIPS-based representation in which actions are limited to quantifier-free, conjunctive preconditions and effects. Since this representation is severely limited, this section discusses extensions to more expressive representations aimed at complex, real-world domains.

Conditional Effects. Conditional effects are used to describe actions whose effects are context-dependent. The basic idea is simple: allow a special *when* clause in the syntax of action effects. *when* takes two arguments, an antecedent and a consequent; execution of the action will have the consequent effect just in the case that the antecedent is true immediately before execution (i.e., much like the action precondition determines if execution itself is legal). Note also that, like an action precondition, the antecedent part refers to the world before the action is executed while the consequent refers to the world after execution. It can be assumed that the consequent is a conjunction of positive or negative literals. Conditional effects are useful when combined with quantification as we will see in the example below.

Universal and Existential Quantification. PDDL allows action schemata with universal and existential quantification. In action effects, only universal quantification is allowed, but goals, preconditions, and conditional effect antecedents may have interleaved universal and existential quantifiers. Quantified formulae are compiled into the corresponding Herbrand base, universal quantification is codified using conjunction while existential is codified using disjunction. Existential quantification is forbidden in action effects because they are equivalent to disjunctive effects and imply non-determinism and hence require reasoning about uncertainty.

As shown in the following example, we can use a universally quantified conditional effects and existentially quantified preconditions to rewrite the action `take` introduced in Section B.1 and avoid the use of a second parameter.

```
(:action take
   :parameters (?x - takeable)
   :precondition
     (not(haslocation ?x myself))
     (exists(?y - container)(haslocation ?x ?y))
   :effect
     (forall(?y - container)(when(haslocation ?x ?y)
                                 (not(haslocation ?x ?y))))
     (haslocation ?x myself))
```

Domain axioms. Axioms are logical formulae that assert relationships among propositions that hold within a situation (as opposed to action definitions, which define relationships across successive situations).

Formally, the syntax for axioms is the following:

```
(:axiom
   :vars ({<variable> - <type>}*)
   :context <assertion>*
   :implies <assertion>*)
```

where the `:vars` field behaves like a universal quantifier. All the variables that occur in the axiom must be declared here.

Action definitions are not allowed to have effects that modify predicates which occur in the `:implies` field of an axiom. The intention is that action definitions mention "primitive" predicates (like `haslocation`), and that all changes in truth value of "derived" predicates (like `accessible`) occur through axioms. Most planners do not verify this restriction syntactically but they do not take responsibility for the outcome due to the complex interactions among actions and axioms.

However, without axioms, the action definitions will have to describe changes in all predicates that might be affected by an action, which leads to a complex "domain engineering" problem.

Even though expressive PDDL can handle more complex domains, few planners will handle the entire PDDL (as we mentioned already in the beginning of this section). The reason for this is that, as expected, the more expressive the language accepted by the planner the higher the computational complexity of the problem that involves finding a plan. Hence, expressive PDDL will not be desirable for all applications; a balance of expressivity and complexity should be found.

Bibliography

[Alahverdzhieva, 2008] Katya Alahverdzhieva. XTAG using XMG. Master's thesis, Université Henri Poincaré, INRIA Nancy Grand Est, France, 2008. Supervised by Claire Gardent. [cited in page: 107]

[Allen and Allen, 1994] James Allen and Richard Allen. *Natural language understanding.* Addison Wesley, 2nd edition, 1994. [cited in page: 25]

[Allen and Core, 1997] James Allen and Mark Core. Draft of DAMSL: Dialog act markup in several layers. Technical report, University of Rochester, 1997. [cited in page: 82]

[Allen and Perrault, 1979] James Allen and Raymond Perrault. Plans, inference, and indirect speech acts. In *Proceedings of the 17th annual meeting of the Association for Computational Linguistics*, pages 85–87, 1979. [cited in page: 70]

[Allen, 1979] James Allen. *A plan-based approach to speech act recognition.* PhD thesis, University of Toronto, Canada, 1979. Supervised by Raymond Perrault. [cited in page: 29, 70]

[Allwood, 1995] Jens Allwood. An activity based approach to pragmatics. In Harry Bunt and William Black, editors, *Abduction, Belief and Context in Dialogue: Studies in Computational Pragmatics*, pages 47–80. John Benjamins Publishers, 1995. [cited in page: 36]

[Asher and Lascarides, 2008] Nicholas Asher and Alex Lascarides. Making the right commitments in dialogue. In *Proceedings of the Fall 2008 Workshop in Philosophy and Linguistics.* University of Michigan, 2008. [cited in page: 149]

[Austin, 1962] John Austin. *How to do Things with Words: The William James Lectures delivered at Harvard University in 1955.* Oxford University Press, 1962. [cited in page: 24, 69]

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003. [cited in page: 98, 99]

[Beaver and Geurts, in press] David Beaver and Bart Geurts. Presupposition. In *Handbook of Semantics.* Mouton de Gruyter, in press. [cited in page: 88, 89]

[Beaver and Zeevat, 2007] David Beaver and Henk Zeevat. Accommodation. In *The Oxford Handbook of Linguistic Interfaces*, pages 503–539. Oxford University Press, 2007. [cited in page: 87]

[Benotti and Traum, 2009] Luciana Benotti and David Traum. A computational account of comparative implicatures for a spoken dialogue agent. In *Proceedings of the Eight International Conference on Computational Semantics*, pages 4–17, Tilburg, The Netherlands, January 2009. [cited in page: 51]

[Benotti, 2006a] Luciana Benotti. "DRINK ME": Handling actions through planning in a text game adventure. In Janneke Huitink and Sophia Katrenko, editors, *Student Session of the European*

Summer School in Logic Language and Information (ESSLLI), pages 160–172, Malaga, Spain, 2006. [cited in page: 125]

[Benotti, 2006b] Luciana Benotti. Enhancing a dialogue system through dynamic planning. Master's thesis, Universidad Politécnica de Madrid, 2006. [cited in page: 125]

[Benotti, 2007] Luciana Benotti. Incomplete knowledge and tacit action: Enlightened update in a dialogue game. In *The 2007 Workshop on the Semantics and Pragmatics of Dialogue (Decalog 2007)*, pages 17–24, Rovereto, Italy, 2007. [cited in page: 132]

[Benotti, 2008a] Luciana Benotti. Accommodation through tacit dialogue acts. In *Conference on Semantics and Modelisation (JSM08)*, Toulouse, France, 2008. [cited in page: 132]

[Benotti, 2008b] Luciana Benotti. Accommodation through tacit sensing. In *The 2008 Workshop on the Semantics and Pragmatics of Dialogue (Londial 2008)*, London, United Kingdom, 2008. [cited in page: 130]

[Benotti, 2009a] Luciana Benotti. Clarification potential of instructions. In *Proceedings of the 2009 SIGDIAL Conference on Discourse and Dialogue*, pages 196–205, London, United Kingdom, September 2009. Best student paper award. [cited in page: 43]

[Benotti, 2009b] Luciana Benotti. Frolog: an accommodating text-adventure game. In *Proceedings of the Demonstrations Session at the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 1–4, Athens, Greece, April 2009. Association for Computational Linguistics. [cited in page: 95]

[Blackburn *et al.*, 2006] Patrick Blackburn, Johan van Benthem, and Frank Wolter. *Handbook of Modal Logic, Volume 3 (Studies in Logic and Practical Reasoning)*. Elsevier Science Inc., New York, USA, 2006. [cited in page: 69]

[Blum and Furst, 1997] Avrim Blum and Merrick Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90(1-2):281–300, 1997. [cited in page: 66]

[Brown and Levinson, 1978] Penelope Brown and Stephen Levinson. *Politeness: Some universals in language usage*. Studies in Interactional Sociolinguistics, 1978. [cited in page: 17, 22, 146, 147]

[Burnard, 2000] Lou Burnard. Reference guide for the British National Corpus (BNC). Technical report, Oxford University Computing Services, 2000. [cited in page: 40]

[Byron *et al.*, 2009] Donna Byron, Alexander Koller, Kristina Striegnitz, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. Report on the 1st NLG Challenge on Generating Instructions in Virtual Environments (GIVE). In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 165–173, Athens, Greece, 2009. [cited in page: 145, 150]

[Charniak and Goldman, 1989] Eugene Charniak and Robert Goldman. A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 1074–1079, 1989. [cited in page: 57]

[Charniak and Shimony, 1990] Eugene Charniak and Solomon Shimony. Probabilistic semantics for cost based abduction. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 106–111, 1990. [cited in page: 57]

[Chemla, 2009] Emmanuel Chemla. Universal implicatures and free choice effects: Experimental data. *Semantics and Pragmatics*, 2(2):1–33, May 2009. [cited in page: 26]

[Clark and Marshall, 1981] Herbert Clark and Catherine Marshall. Definite reference and mutual knowledge. In A. Joshi, B. Webber, and I. Sag, editors, *Elements of discourse understanding*, pages 10–63. Cambridge University Press, 1981. [cited in page: 89]

[Clark and Schaefer, 1989] Herbert Clark and Edward Schaefer. Contributing to discourse. *Cognitive Science*, 13(2):259–294, 1989. [cited in page: 29]

[Clark and Wilkes-Gibbs, 1986] Herbert Clark and Deanna Wilkes-Gibbs. Referring as a collaborative process. *Cognition*, 22:1–39, 1986. [cited in page: 26, 73, 83, 84]

[Clark, 1975] Herbert Clark. Bridging. In *Proceedings of the 1975 workshop on Theoretical issues in natural language processing (TINLAP 75)*, pages 169–174, Morristown, USA, 1975. Association for Computational Linguistics. [cited in page: 88, 142]

[Clark, 1996] Herbert Clark. *Using Language*. Cambridge University Press, New York, 1996. [cited in page: 12, 36, 89, 93, 114, 149, 151]

[Cohen and Perrault, 1979] Philip Cohen and Raymond Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3):177 – 212, 1979. [cited in page: 70]

[Crabbé and Duchier, 2005] Benoit Crabbé and Denys Duchier. Metagrammar redux. In *Constraint Solving and Language Processing*, volume 3438 of *Lecture Notes in Computer Science*, pages 32–47. Springer, 2005. [cited in page: 106]

[Davidson, 1967] Donald Davidson. The logical form of action sentences. In Nicholas Rescher, editor, *The Logic of Decision and Action*. University of Pittsburgh Press, 1967. [cited in page: 81]

[DeVault and Stone, 2009] David DeVault and Matthew Stone. Learning to interpret utterances using dialogue history. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 184–192, Athens, Greece, 2009. Association for Computational Linguistics. [cited in page: 145]

[DeVault *et al.*, 2006] David DeVault, Iris Oved, and Matthew Stone. Societal grounding is essential to meaningful language use. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2006. [cited in page: 124, 147]

[DeVault, 2008] David DeVault. *Contribution Tracking: Participating in Task-Oriented Dialogue under Uncertainty*. PhD thesis, Department of Computer Science, Rutgers, The State University of New Jersey, New Brunswick, USA, 2008. Supervised by Matthew Stone. [cited in page: 130, 144]

[Eshghi, 1988] Kave Eshghi. Abductive planning with event calculus. In *Proceedings of the International Conference in Logic Programming*, pages 562–579, 1988. [cited in page: 58, 70, 143]

[Estlin *et al.*, 2003] Tara Estlin, Rebecca Castano, Robert Anderson, Daniel Gaines, Forest Fisher, and Michele Judd. Learning and planning for Mars Rover Science. In *Proceedings of the IJCAI Workshop on Issues in Designing Physical Agents for Dynamic Real-Time Environments: World Modeling, Planning, Learning, and Communicating*. Morgan Kaufmann Publishers, 2003. [cited in page: 59]

[Fikes *et al.*, 1972] Richard Fikes, Peter Hart, and Nils Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288, 1972. [cited in page: 62, 74, 112]

[Foster and Matheson, 2008] Mary Ellen Foster and Colin Matheson. Following assembly plans in cooperative, task-based human-robot dialogue. In *Proceedings of the 12th Workshop on the Semantics and Pragmatics of Dialogue (Londial 2008)*, London, June 2008. [cited in page: 93]

[Foster *et al.*, 2009] Mary Ellen Foster, Manuel Giuliani, Amy Isard, Colin Matheson, Jon Oberlander, and Alois Knoll. Evaluating description and reference strategies in a cooperative human-robot dialogue system. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09)*, Pasadena, California, July 2009. [cited in page: 93]

[Gabsdil, 2003] Malte Gabsdil. Clarification in spoken dialogue systems. In *Proceedings of the AAAI Spring Symposium on Natural Language Generation in Spoken and Written Dialogue*, pages 28–35, Palo Alto, California, 2003. [cited in page: 34, 36, 38, 39, 45]

[Gardent and Kow, 2007] Claire Gardent and Eric Kow. A symbolic approach to near-deterministic surface realisation using tree adjoining grammar. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 328–335, Prague, Czech Republic, 2007. [cited in page: 107, 109]

[Gardent, 2008] Claire Gardent. Integrating a unification-based semantics in a large scale lexicalised tree adjoininig grammar for french. In *Proceedings of the International Conference on Computational Linguistics (COLING 2008)*, 2008. [cited in page: 106]

[Gerevini and Long, 2005] Alfonso Gerevini and Derek Long. Plan constraints and preferences in PDDL3. Technical Report R.T. 2005-08-47, Università degli Studi di Brescia, Italy, 2005. [cited in page: 64, 154, 159]

[Geurts and Pouscoulous, 2009] Bart Geurts and Nausicaa Pouscoulous. Embedded implicatures?!? *Semantics and Pragmatics*, 2(4):1–34, July 2009. [cited in page: 26]

[Geurts, 1999] Bart Geurts. *Presuppositions and Pronouns*, volume 3 of *Current Research in the Semantics/pragmatics Interfaces*. Elsevier, Amsterdam, 1999. [cited in page: 89]

[Geurts, in press] Bart Geurts. *Quantity implicatures*. Cambridge University Press, in press. [cited in page: 20, 24, 143, 151]

[Ginzburg, in press] Jonathan Ginzburg. *The interactive Stance: Meaning for Conversation*. CSLI Publications, in press. [cited in page: 34, 35, 36, 40, 147]

[Goldman *et al.*, 1999] Robert Goldman, Christopher Geib, and Christopher Miller. A new model of plan recognition. *Artificial Intelligence*, 64:53–79, 1999. [cited in page: 67]

[Grice, 1975] Paul Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics*, volume 3, pages 41–58. Academic Press, New York, 1975. [cited in page: 17, 18, 20, 24, 84, 148]

[Grosz and Sidner, 1990] Barbara Grosz and Candace Sidner. Plans for discourse. In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Plans and Communication*, pages 417–444. MIT Press, Cambridge, Massachusetts, 1990. [cited in page: 29]

[Group, 2001] XTAG Research Group. A lexicalized tree adjoining grammar for english. Technical Report IRCS-01-03, University of Pennsylvania, 2001. [cited in page: 107]

[Gundel and Fretheim, 2004] Jeanette Gundel and Thorstein Fretheim. Topic and focus. In *Handbook of Pragmatics*, pages 241–265. Blackwell, Oxford, 2004. [cited in page: 109]

[Haarslev and Möller, 2001] Volker Haarslev and Ralf Möller. RACER system description. In *Proceedings of International Joint Conference on Automated Reasoning (IJCAR-01)*, pages 701–705, Siena, Italy, 2001. [cited in page: 98, 109]

[Heim, 1982] Irene Heim. *On the Semantics of Definite and Indefinite Noun Phrases*. PhD thesis, University of Massachusetts at Amherst, 1982. [cited in page: 88]

[Hobbs *et al.*, 1990] Jerry Hobbs, Mark Stickel, Douglas Appelt, and Paul Martin. Interpretation as abduction. *Artificial Intelligence*, 63:69–142, 1990. [cited in page: 57, 58]

[Hobbs, 1985] Jerry Hobbs. Granularity. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85)*, pages 432–435. Morgan Kaufmann, 1985. [cited in page: 10]

[Hobbs, 2004] Jerry Hobbs. Abduction in natural language understanding. In *Handbook of Pragmatics*, pages 724–741. Blackwell, Oxford, 2004. [cited in page: 24]

[Hobbs, to appear] Jerry Hobbs. Discourse and inference: Magnum opus in progress. Obtained from Jerry Hobbs website in July 2009, to appear. [cited in page: 58]

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *JAIR*, 14:253–302, 2001. [cited in page: 146]

[Hoffmann *et al.*, 2004] Jörg Hoffmann, Julie Porteous, and Laura Sebastia. Ordered landmarks in planning. *Journal of Artificial Intelligence Research*, 22:215–278, 2004. [cited in page: 146]

[Horn, 2004] Larry Horn. Implicature. In *Handbook of Pragmatics*, pages 3–28. Blackwell, Oxford, 2004. [cited in page: 18]

[Huber *et al.*, 1994] Marcus Huber, Edmund Durfee, and Michael Wellman. The automated mapping of plans for plan recognition. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 344–351. Morgan Kaufmann, 1994. [cited in page: 67]

[Jurafsky and Martin, 2008] Daniel Jurafsky and James Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Prentice Hall, 2008. [cited in page: 25]

[Jurafsky, 2004] Daniel Jurafsky. Pragmatics and computational linguistics. In *Handbook of Pragmatics*, pages 3–28. Blackwell, Oxford, 2004. [cited in page: 22, 25, 70, 82, 143]

[Kallmeyer *et al.*, 2008] Laura Kallmeyer, Timm Lichte, Wolfgang Maier, Yannick Parmentier, Johannes Dellert, and Kilian Evang. TuLiPA: Towards a multi-formalism parsing environment for grammar engineering. *Computing Research Repository of the ACM (CoRR)*, 0807.3622, 2008. [cited in page: 107]

[Kautz and Selman, 1992] Henry Kautz and Bart Selman. Planning as satisfiability. In *Proceedings of the 10th European conference on Artificial intelligence (ECAI 92)*, pages 359–363, New York, USA, 1992. John Wiley & Sons, Inc. [cited in page: 66]

[Kautz and Selman, 1999] Henry Kautz and Bart Selman. Unifying SAT-based and graph-based planning. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 318–325, Stockholm, Sweden, 1999. [cited in page: 99, 121, 125, 145]

[Kautz, 1991] Henry Kautz. A formal theory of plan recognition and its implementation. In *Reasoning about plans*, pages 69–124. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991. [cited in page: 67]

[Kehler, 2004] Andrew Kehler. Discourse coherence. In *Handbook of Pragmatics*, pages 241–265. Blackwell, Oxford, 2004. [cited in page: 22]

[Kelley *et al.*, 2008] Richard Kelley, Alireza Tavakkoli, Christopher King, Monica Nicolescu, Mircea Nicolescu, and George Bebis. Understanding human intentions via hidden markov models in autonomous mobile robots. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction (HRI 08)*, pages 367–374, New York, NY, USA, 2008. ACM. [cited in page: 67]

[Kenny, 1966] Anthony Kenny. Practical inference. *Analysis*, 26:65–75, 1966. [cited in page: 53, 146]

[Koehler and Hoffmann, 2000] Jana Koehler and Jörg Hoffmann. On reasonable and forced goal orderings and their use in an agenda-driven planning algorithm. *Journal of Artificial Intelligence Research*, 12:338–386, 2000. [cited in page: 66]

[Koller *et al.*, 2004] Alexander Koller, Ralph Debusmann, Malte Gabsdil, and Kristina Striegnitz. Put my galakmid coin into the dispenser and kick it: Computational linguistics and theorem proving in a computer game. *Journal of Logic, Language and Information*, 13(2):187–206, 2004. [cited in page: 97, 110]

[Kow *et al.*, 2006] Eric Kow, Yannick Parmentier, and Claire Gardent. SemTAG, the LORIA toolbox for TAG-based parsing and generation. In *Proceedings of the 8th International Workshop on Tree Adjoining Grammar and Related Formalisms*, pages 115–120, Sydney, Australia, July 2006. Association for Computational Linguistics. [cited in page: 107]

[Kreutel and Matheson, 2003] Jörn Kreutel and Colin Matheson. Context-dependent interpretation and implicit dialogue acts. In *Perspectives on Dialogue in the New Millenium*, pages 179–192. John Benjamins, 2003. [cited in page: 132, 144]

[Larsson and Traum, 2000] Staffan Larsson and David Traum. Information state and dialogue management in the trindi dialogue move engine toolkit. *Natural Language Engineering*, 6(3-4):323–340, 2000. [cited in page: 144]

[Larsson, 2007] Staffan Larsson. Coordinating on ad-hoc semantic systems in dialogue. In *The 2007 Workshop on the Semantics and Pragmatics of Dialogue (Decalog 2007)*, 2007. [cited in page: 124, 147]

[Levesque, 1996] Hector Levesque. What is planning in the presence of sensing? In *Proceedings of the 14th Conference on Artificial Intelligence*, pages 1139–1146. AAAI Press, 1996. [cited in page: 68, 130, 133]

[Levinson, 1983] Stephen Levinson. *Pragmatics*. Cambridge textbooks in linguitics, 1983. [cited in page: 16, 18, 19, 22, 23, 51, 87, 141, 142, 149]

[Levinson, 2000] Stephen Levinson. *Presumptive Meanings*. MIT Press, Cambridge, 2000. [cited in page: 27]

[Lewis, 1979] David Lewis. Scorekeeping in a language game. *Journal of Philosophical Logic*, 8:339–359, 1979. [cited in page: 87, 131, 142]

[Mills, 2007] Gregory Mills. *Semantic co-ordination in dialogue: the role of direct interaction*. PhD thesis, Queen Mary, University of London, 2007. Supervised by Patrick Healey. [cited in page: 83, 124]

[Nau *et al.*, 2004] Dana Nau, Malik Ghallab, and Paolo Traverso. *Automated Planning: Theory & Practice*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. [cited in page: 58, 61, 123]

[Norvig and Wilensky, 1990] Peter Norvig and Robert Wilensky. A critical evaluation of commensurable abduction models for semantic interpretation. In *Proceedings of the 13th conference on Computational linguistics*, pages 225–230, Morristown, NJ, USA, 1990. Association for Computational Linguistics. [cited in page: 58, 70]

[Parmentier, 2007] Yannick Parmentier. *SemTAG : une plate-forme pour le calcul sémantique à partir de Grammaires d'Arbres Adjoints*. PhD thesis, Université Henri Poincaré, INRIA Nancy Grand Est, France, 2007. Supervised by Claire Gardent. [cited in page: 107]

[Pearl, 1985] Judea Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society*, pages 329–334, University of California, August 1985. [cited in page: 57]

[Peirce, reprinted in 1955] Charles Peirce. Abduction and induction. In Justus Buchler, editor, *Philosophical writings of Pierce*, pages 150–156. New York: Dover Books, reprinted in 1955. [cited in page: 24, 55]

[Petrick and Bacchus, 2002] Ronald Petrick and Fahiem Bacchus. A knowledge-based approach to planning with incomplete information and sensing. In Malik Ghallab, Joachim Hertzberg, and Paolo Traverso, editors, *Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2002)*, pages 212–221, Menlo Park, CA, 2002. AAAI Press. [cited in page: 130, 132]

[Petrick and Bacchus, 2004] Ronald Petrick and Fahiem Bacchus. Extending the knowledge-based approach to planning with incomplete information and sensing. In Didier Dubois, Christopher Welty, and Mary-Anne Williams, editors, *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR-2004)*, pages 613–622, Menlo Park, CA, 2004. AAAI Press. [cited in page: 68, 99, 132, 146]

[Petrick, 2006] Ronald Petrick. *A Knowledge-level approach for effective acting, sensing, and planning.* PhD thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, 2006. Supervised by Hector Levesque. [cited in page: 68, 145]

[Potts, 2007] Christopher Potts. Conventional implicatures, a distinguished class of meanings. In Gillian Ramchand and Charles Reiss, editors, *The Oxford Handbook of Linguistic Interfaces*, Studies in Theoretical Linguistics, pages 475–501. Oxford University Press, Oxford, 2007. [cited in page: 18, 19]

[Purver *et al.*, 2003] Matthew Purver, Jonathan Ginzburg, and Patrick Healey. On the means for clarification in dialogue. In *Current and New Directions in Discourse and Dialogue*, pages 235–255. Kluwer Academic Publishers, 2003. [cited in page: 39]

[Purver, 2004] Matthew Purver. *The Theory and Use of Clarification Requests in Dialogue.* PhD thesis, King's College, University of London, 2004. Supervised by Jonathan Ginzburg. [cited in page: 34, 35, 40, 43]

[Purver, 2006] Matthew Purver. CLARIE: Handling clarification requests in a dialogue system. *Research on Language and Computation*, 4(2-3):259–288, 2006. [cited in page: 50]

[Quine, 1970] Willard Van Orman Quine. *Philosophy of logic.* Prentice-Hall, 1970. [cited in page: 19]

[Rieser and Moore, 2005] Verena Rieser and Johanna Moore. Implications for generating clarification requests in task-oriented dialogues. In *Proc of ACL*, pages 239–246, 2005. [cited in page: 25, 34, 35, 36, 40, 41, 43, 147]

[Rodríguez and Schlangen, 2004] Kepa Rodríguez and David Schlangen. Form, intonation and function of clarification requests in german task oriented spoken dialogues. In *Proc of SEMDIAL*, pages 101–108, 2004. [cited in page: 34, 35, 36, 41, 147, 151]

[Schegloff, 1987a] Emanuel Schegloff. Between micro and macro: Contexts and other connections. In *Micro-Macro Link*, pages 207–234. University of California Press, 1987. [cited in page: 147]

[Schegloff, 1987b] Emanuel Schegloff. Some sources of misunderstanding in talk-in-interaction. *Linguistics*, 8:201–218, 1987. [cited in page: 34, 147]

[Schlangen and Fernández, 2007] David Schlangen and Raquel Fernández. Beyond repair–testing the limits of the conversational repair system. In *Proceedings of the 2007 SIGDIAL Workshop on Discourse and Dialogue*, pages 51–54, Antwerp, Belgium, 2007. [cited in page: 28, 40]

[Schlangen, 2004] David Schlangen. Causes and strategies for requesting clarification in dialogue. In *Proceedings of the 2004 SIGDIAL Workshop on Discourse and Dialogue*, 2004. [cited in page: 43]

[Schlenker, 2007] Philippe Schlenker. Anti-dynamics: Presupposition projection without dynamic semantics. *Journal of Logic, Language and Information*, 16(3), 2007. [cited in page: 89]

[Schmidt *et al.*, 1978] Charles Schmidt, N. S. Sridharan, and John Goodson. The plan recognition problem: an intersection of psychology and artificial intelligence. *Artificial Intelligence*, 11(17):45–82, 1978. [cited in page: 67]

[Searle, 1975] John Searle. Indirect speech acts. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics*, volume 3, pages 59–82. Academic Press, San Diego, CA, 1975. [cited in page: 25]

[Skantze, 2007] Gabriel Skantze. *Error Handling in Spoken Dialogue Systems*. PhD thesis, KTH - Royal Institute of Technology, Sweden, 2007. Supervised by Rolf Carlson. [cited in page: 34]

[Stalnaker, 1998] Robert Stalnaker. On the representation of context. *Journal of Logic, Language and Information*, 7(1):3–19, 1998. [cited in page: 131]

[Steedman and Petrick, 2007] Mark Steedman and Ronald Petrick. Planning dialog actions. In *Proceedings of the 2007 SIGDIAL Workshop on Discourse and Dialogue*, pages 265–272, Antwerp, Belgium, 2007. [cited in page: 70]

[Steedman, 2002] Mark Steedman. Plans, affordances, and combinatory grammar. *Linguistics and Philosophy*, 25(5-6), 2002. [cited in page: 111]

[Stoia *et al.*, 2008] Laura Stoia, Darla Shockley, Donna Byron, and Eric Fosler-Lussier. SCARE: A situated corpus with annotated referring expressions. In *Proceedings of The International conference on Language Resources and Evaluation (LREC)*, 2008. [cited in page: 42, 151]

[Stoia, 2007] Laura Stoia. *Noun Phrase Generation for Situated Dialogs*. PhD thesis, Ohio State University, USA, 2007. Supervised by Donna Byron. [cited in page: 44, 153]

[Strawson, 1964] Peter Strawson. Identifying reference and truth values. *Theoria*, 30, 1964. [cited in page: 89, 91]

[Thomason *et al.*, 2006] Richmond Thomason, Matthew Stone, and David DeVault. Enlightened update: A computational architecture for presupposition and other pragmatic phenomena. In Donna Byron, Craige Roberts, and Scott Schwenter, editors, *Presupposition Accommodation*. Ohio State Pragmatics Initiative, 2006. draft version 1.0. [cited in page: 86, 87, 120, 132, 144]

[Thomason, 1990] Richmond Thomason. Accommodation, meaning, and implicature: Interdisciplinary foundations for pragmatics. In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*, pages 326–363. MIT Press, Cambridge, Massachusetts, 1990. [cited in page: 20, 29, 84, 148]

[Traum, 1994] David Traum. *A Computational Theory of Grounding in Natural Language Conversation*. PhD thesis, Computer Science Dept., U. Rochester, USA, 1994. Supervised by James Allen. [cited in page: 130]

[Traum, 2003] David Traum. Semantics and pragmatics of questions and answers for dialogue agents. In *Proceedings of the International Workshop on Computational Semantics (IWCS)*, pages 380–394, January 2003. [cited in page: 39]

[van der Sandt, 1992] Rob van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9:333–377, 1992. [cited in page: 89]

[Wilson and Sperber, 2004] Deirdre Wilson and Dan Sperber. Relevance theory. In *Handbook of Pragmatics*, pages 607–632. Blackwell, Oxford, 2004. [cited in page: 21, 148]

[Zacks and Tversky, 2001] Jeff Zacks and Barbara Tversky. Event structure in perception and conception. *Psychological Bulletin*, 127:3–21, 2001. [cited in page: 11, 14]

Index