# A Dynamic Logic Formalisation of the Dialogue Gameboard

**Raquel Fernández**
Department of Computer Science
King's College London
raquel@dcs.kcl.ac.uk

## Abstract

This paper explores the possibility of using the paradigm of Dynamic Logic (DL) to formalise information states and update processes on information states. In particular, we present a formalisation of the *dialogue gameboard* introduced by Jonathan Ginzburg. From a more general point of view, we show that DL is particularly well suited to develop rigorous formal foundations for an approach to dialogue dynamics based on information state updates.

## 1 Introduction

A particular development that has received much attention in recent work on dialogue modelling is the use of *information states* to characterise the state of each dialogue participant's information as the conversation proceeds. The information state approach to dialogue, as developed for instance in the TRINDI project (e.g. (Bohlin et al., 1999; Traum et al., 1999)), assumes that some aspects of dialogue management are best captured in terms of the relevant information that is available to each dialogue participant at each state of the conversation, along with a full account of the possible update mechanisms that change this information. Unlike classical Artificial Intelligence approaches built on the basis of axiomatic theories of rational agency,[1] information state accounts tend to avoid the use of logical frameworks and concentrate on dialogue-specific notions such as common ground, discourse obligations and questions under discussion.

In this paper we explore the possibility of using a modal logic paradigm, namely Dynamic Logic (Harel et al., 2000), originally conceived as a formal system to reason about computer programs, to formalise information states and update processes on information states. In particular, we present a dynamic logic formalisation of Ginzburg's *dialogue gameboard* (DGB) as introduced in (Ginzburg, 1996; Ginzburg, ms) and (Larsson, 2002). From a more general point of view, we show that Dynamic Logic is particularly well suited to develop rigorous formal foundations for an approach to dialogue dynamics based on information state updates.

### 1.1 Overview

The structure of the paper is as follows: First, we introduce the basic notions of First-Order Dynamic Logic, describing its syntax and semantics. After briefly characterising the structure of the dialogue gameboard in Section 3, our formalisation is presented in Section 4. We define the formal language and its semantic interpretation, and discuss how the different components of the dialogue gameboard have been modelled. In Section 5, we show how the rules of conversational interaction can be expressed within the formalism and explain some examples in detail. Finally, in Section 6, we present our conclusions and indicate some directions for future research.

[1] See e.g. (Cohen and Levesque, 1990; Grosz and Sidner, 1990; Sadek, 1991).

| | | |
|---|---|---|
| $\mathcal{M} \models_s \varphi$ | iff | $\mathcal{A} \models \varphi[v]_s$, for atomic formulae $\varphi$ |
| $\mathcal{M} \models \top$ | | $\top$ is always true |
| $\mathcal{M} \not\models \bot$ | | $\bot$ is never true |
| $\mathcal{M} \models_s (t_1 = t_2)$ | iff | $v_s(t_1)$ equals $v_s(t_2)$, for terms $t_1$ and $t_2$ |
| $\mathcal{M} \models_s \neg A$ | iff | $\mathcal{M} \not\models_s A$ |
| $\mathcal{M} \models_s (A_1 \wedge A_2)$ | iff | $\mathcal{M} \models_s A_1$ and $\mathcal{M} \models_s A_2$ |
| $\mathcal{M} \models_s (A_1 \vee A_2)$ | iff | $\mathcal{M} \models_s A_1$ or $\mathcal{M} \models_s A_2$ |
| $\mathcal{M} \models_s (A_1 \rightarrow A_2)$ | iff | $\mathcal{M} \not\models_s A_1$ or $\mathcal{M} \models_s A_2$ |
| $\mathcal{M} \models_s \exists x A$ | iff | there is an $a \in D$, such that $s(x\|a)s'$ and $\mathcal{M} \models_{s'} A$ |
| $\mathcal{M} \models_s \forall x A$ | iff | for all $a \in D$, if $s(x\|a)s'$ then $\mathcal{M} \models_{s'} A$ |
| $\mathcal{M} \models_s <\alpha> A$ | iff | there is an $s' \in S$, such that $sR_\alpha s'$ and $\mathcal{M} \models_{s'} A$ |
| $\mathcal{M} \models_s [\,\alpha\,] A$ | iff | for all $s' \in S$, if $sR_\alpha s'$ then $\mathcal{M} \models_{s'} A$ |

Table 1: Definition of truth

| | | |
|---|---|---|
| $sR_{x:=t}s'$ | iff | $s(x\|v_s(t))s'$ |
| $sR_{\alpha;\beta}s'$ | iff | $\exists s''$ such that $sR_\alpha s''$ and $s''R_\beta s'$ |
| $sR_{\alpha\cup\beta}s'$ | iff | $sR_\alpha s'$ or $sR_\beta s'$ |
| $sR_{\alpha*}s'$ | iff | there are finitely many states $s_1, s_2, \ldots, s_n$ such that $s_1R_\alpha s_2, s_2R_\alpha s_3, \ldots, s_{n-1}R_\alpha s_n$ and $s = s_1$ and $s' = s_n$ |
| $sR_{\varphi?}s'$ | iff | $s = s'$ and $\mathcal{M} \models_s \varphi$ |

Table 2: Accessibility relations

## 2 Dynamic Logic: Basic Notions

The formalisation we present in this paper is based on the first-order version of Dynamic Logic (DL) as it is discussed in (Harel et al., 2000) and (Goldblatt, 1992). In short, DL is a multi-modal logic with a possible worlds semantics, which distinguishes between expressions of two sorts: *formulae* and *programs*. The language of DL is that of first-order logic together with a set of modal operators: for each program $\alpha$ there are a box $[\,\alpha\,]$ and a diamond $<\alpha>$ operator. The set of possible worlds (or states) in the model is the set of all possible assignments to the variables in the language. Atomic programs change the values assigned to particular variables. They can be combined to form complex programs by means of a repertoire of program constructs, such as *sequence* ; , *non-deterministic choice* $\cup$, *iteration* * and *test* ?.

Originally, DL was conceived as a formal system to reason about programs, formalising correctness specifications and proving rigorously that those specifications are met by a particular program. From a more general perspective, however, it can be viewed as a formal system to reason about transformations on states. In this sense, it is particularly well suited to provide a fine characterisation of the dynamic processes that take place in dialogue as updates on the information states of the dialogue participants.

In the remainder of this section, we formally introduce the syntax and the semantics of DL.

### 2.1 Syntax

The language of first-order DL is built upon First-Order Logic. It is generated by some first-order vocabulary $\Sigma$ made up of a set of predicate symbols, a set of function symbols, a set of constants and a set of variables. In addition to the propositional connectives and the universal and existential quantifier symbols, the language also includes two modal operators [ ] and <>, a set $\Pi$ of programs $\alpha$ and the program constructs ;, $\cup$, * and ?.

**Formulae and Programs.** Atomic formulae $\varphi$ are atomic, first-order formulae of the vocabulary $\Sigma$, including $\top$ and $\bot$. The set $\Phi$ of well-formed

formulae $A$ is then defined as follows:

$$A ::= \varphi \mid \neg A \mid A_1 \wedge A_2 \mid A_1 \vee A_2 \mid A_1 \rightarrow A_2 \mid \forall x A \mid \exists x A \mid [\alpha]\, A \mid <\alpha> A$$

In the basic version of DL, atomic programs $\pi$ are simple assignments ($x := t$), where $x$ is an individual variable and $t$ is a first-order term. The set $\Pi$ of programs $\alpha$ is defined as follows:

$$\alpha ::= \pi \mid \alpha_1; \alpha_2 \mid \alpha_1 \cup \alpha_2 \mid \alpha^* \mid \varphi?$$

### 2.2 Semantics

As usual in modal logic, the language is interpreted in a possible-worlds based semantical structure. A model is a structure

$$\mathcal{M} = \{\mathcal{A}, S, R, V\}$$

where

- $\mathcal{A} = \{D, I\}$ is a first-order structure;
- $S$ is a non-empty set of states;
- $R$ is a function assigning to each program $\alpha \in \Pi$ a binary relation $R_\alpha \subseteq S \times S$;
- $V$ is a function $V : S \rightarrow S^{\mathcal{A}}$ assigning to each $s \in S$ an $\mathcal{A}$-valuation $v_s : \mathit{Var} \rightarrow D$, i.e. a mapping from the set of variables to elements in the domain.

For $s, s' \in S$, we will write $s(x|a)s'$ to mean that $v_{s'}(x) = a$ and $v_{s'}(y) = v_s(y)$ whenever $y \neq x$.

Now we are ready to define the *truth-relation* $\mathcal{M} \models_s A$ of a formula $A$ at state $s$ in model $\mathcal{M}$. As usual in first-order logic, we write $\mathcal{A} \models \varphi[v]$ to mean that $\varphi$ is *true in* $\mathcal{A}$ *under valuation* $v$. For conciseness, we will omit the part dealing with the semantics of first-order terms. The formal definition of truth in a model is shown in Table 1.

From the relations $R_\alpha \subseteq S \times S$, we can inductively define accessibility relations for the compound programs. Table 2 shows the accessibility relations for basic atomic programs and compound programs for all states $s, s' \in S$.

**Stack Variables.** Interesting variants of DL arise from allowing auxiliary data structures such as *stacks* and *arrays*. Following (Harel et al., 2000), we will consider a version of DL in which programs can manipulate some variables as last-in-first-out stacks. Formally, stacks are modelled as variables ranging over finite strings of elements in the domain. To manipulate these *stack variables*, two additional atomic programs $\mathtt{X}.\mathbf{pop}$ and $\mathtt{X}.\mathbf{push}(x)$ are included. Here $\mathtt{X}$ is some stack variable (i.e. a string of elements ) and $x$ stands for the element to be pushed onto $\mathtt{X}$. The accessibility relations for these two new atomic programs are shown in Table 3, where, for a string $\sigma$ and an element $a$, $\mathit{tail}(a \cdot \sigma) = \sigma$.

| | | |
|---|---|---|
| $sR_{\mathtt{X}.\mathbf{push}(x)}s'$ | iff | $s(\mathtt{X} \mid v_s(x) \cdot v_s(\mathtt{X}))s'$ |
| $sR_{\mathtt{X}.\mathbf{pop}}s'$ | iff | $s(\mathtt{X} \mid \mathit{tail}(v_s(\mathtt{X}))s'$ |

Table 3: **push** and **pop** programs

## 3 The Dialogue Gameboard

Following the pioneering work of philosophers like (Lewis, 1979) and (Stalnaker, 1979), the theory of context developed by Jonathan Ginzburg joins a line of research which, instead of focusing on the intentional attitudes of the dialogue participants, highlights the public and conventional aspects of communication. Under this perspective, a dialogue can be thought of as a *conversational scoreboard* that keeps track of the state of the conversation.

The *dialogue gameboard* (DGB), Ginzburg's particular version of the *conversational scoreboard*, plays a central role in his theory of context. It can be seen as the context relative to which conventionalised interaction is assumed to take place. The DGB provides a structured characterisation of the information which the dialogue participants view as common in terms of three main components: a set of FACTS, which the dialogue participants take as common ground, a partially ordered set of questions under discussion QUD, and the LATEST-MOVE made in the dialogue. Inspired by the notion of *dialogue game* (e.g. (Hamblin, 1970; Carlson, 1983)), Ginzburg assumes that each move made by a dialogue participant determines a restricted set of options for follow-up in the dialogue, constraining what can be said and how.

The framework has been used to provide an account of the kind of context that licenses elliptical responses in dialogue (Ginzburg, 1999; Fernández

and Ginzburg, 2002; Fernández et al., 2003) and has also been the starting point of implemented dialogue systems such as GoDiS (Cooper et al., 2001) and IBiS (Larsson, 2002).

## 4 A DL Formalisation of the DGB

To model context in dialogue as it is understood in Ginzburg's DGB, we will consider a particular domain of interpretation which includes entities such as agents (the dialogue participants), questions, propositions and dialogue moves.[2] For the sake of simplicity, in this paper we restrict ourselves to four dialogue move types, namely *ask, assert, clarification request* and *acknowledge*. The main strategy to reason about the effects of conversational interaction on the DGB, will be to represent its main components as variables ranging over different domains. In what follows, we introduce the details of our formalism.

### 4.1 Introducing the Formalism

Let $\mathcal{L}$ be a first-order DL language with equality made up of unary predicate symbols $Q, P, G, DP$, binary predicate symbols **infl**(uences) and **ans**(wers), a ternary predicate symbol $Utt$, a function symbol **whether**, constants $a, b$, `ask`, `ass`, `clr` and `ack`, and an infinite set *Var* of variables $x$. *Var* includes a set $V_1 = \{\texttt{LM}_a, \texttt{LM}_b, \texttt{UTT}\}$ of special individual variables and a set $V_2 = \{\texttt{FACTS}, \texttt{QUD}_a, \texttt{QUD}_b, \texttt{PENDING}_a, \texttt{PENDING}_b\}$ of stack variables. We also introduce a function symbol **head** to be applied to stack variables.

The set of variable symbols *Var* also includes symbols $i, j$ which range over the set of dialogue participants, symbols $q, q'$ and $p, p'$ ranging over questions and propositions respectively, symbols $r, r'$ ranging over propositions or questions, symbols $m, m'$ ranging over moves, and symbols $u, u'$ ranging over utterances.

Language $\mathcal{L}$ is interpreted over a first-order structure $\mathcal{A} = \{D, \mathcal{I}\}$. The domain $D$ of $\mathcal{A}$ is made up of a set of dialogue participants $DP^{\mathcal{D}} = \{a^{\mathcal{I}}, b^{\mathcal{I}}\}$, a set of questions $Q^{\mathcal{D}}$, a set of propositions $P^{\mathcal{D}}$, a set of dialogue moves $M = \{\texttt{ask}^{\mathcal{I}}, \texttt{ass}^{\mathcal{I}}, \texttt{clr}^{\mathcal{I}}, \texttt{ack}^{\mathcal{I}}\}$, and an element 1 which is used to interpret the predicate symbol $G$, i.e. we set $\mathcal{I}(G) = \{1\}$. A number of relations are declared over $D$: **infl** is interpreted as a binary relation on $Q^{\mathcal{D}}$, **ans** as a binary relation between $P^{\mathcal{D}}$ and $Q^{\mathcal{D}}$, and $Utt$ as a set of utterances $Utt^{\mathcal{D}}$, that will be modelled as triples $(i, m, r)$ of a dialogue participant, a dialogue move and either a proposition or a question. The function symbol **whether** is interpreted as a function $whether$ such that for every proposition $p$, $whether(p) \in Q^{\mathcal{D}}$. Finally, **head** is interpreted as a function that maps every string to its first element.

Recall that stack variables range over strings of elements in the domain: Let $Q^*$, $P^*$, $Utt^*$ denote the set of all finite-length strings over $Q^{\mathcal{D}}$, $P^{\mathcal{D}}$ and $Utt^{\mathcal{D}}$, respectively. This will be used later on to model the stack variables in $V_2$.

### 4.2 The DGB Components

As mention earlier, in DL, transitions between states are changes in variable assignment. We therefore represent the dynamic aspects of the information state as variables ranging over different domains. In particular, we use the variable names `FACTS`, `QUD` and `LM` to represent the three different components of the DGB. We also include two additional variables `UTT` and `PENDING`. New utterances are assigned to `UTT` and, in case the addressee cannot ground their content, they are also assigned to `PENDING`. This allows to distinguish between two kinds of grounding: content grounding (the value of `UTT` is assigned to `LM`) and proposition grounding or acceptance (a proposition is incorporated onto `FACTS`).

To model content grounding we use a unary predicate $G$ and assume that $G(x)$ only holds when the addressee of a particular utterance can ground its content. That is, according to the formalisation introduced in Section 4.1, $G(x)$ will be true in all those states where $v(x) = 1$. As an abbreviation, we will write $G$ when $G(x)$ and $v(x) = 1$, and $\neg G$ otherwise.

One of the assumptions behind the DGB is that a realistic characterisation of context must allow for asymmetries between the information available to the different dialogue participants at a given point in a conversation. Thus, although the

[2] Note that both propositions and questions are first-class entities in the domain. While this is not the standard approach, it is familiar from situation theoretic work and makes the current formalisation simpler.

DGB attempts to represent the publicly accessible information at each state of the dialogue, it does so in terms of the collection of individual information states of the participants. In the current formalisation, however, only QUD, LM and PENDING are relative to each dialogue participant, while FACTS and UTT are unique. This is an obvious choice for the case of UTT, which is just used to hold new contributions publicly uttered by any dialogue participant. In the case of FACTS, however, this is a simplification motivated by the fact that the current formalisation only attempts to model simplified situations where FACTS is assumed to be empty at the initial state, and only propositions that have been commonly agreed on can be integrated into it. Thus, there is no room for disagreements in this respect, and the set of FACTS is always the same for the two dialogue participants.

We model QUD and PENDING as stacks, in a way that is very much inspired by QUD's actual implementation in the GoDiS dialogue system (Cooper et al. 2001). Although we think of FACTS as a set,[3] for technical reasons that will become clear below, we also model FACTS as a stack. On the other hand, UTT and LM range over utterances, i.e. triples $(i, m, r)$, where $i$ is interpreted as the speaker of $u$, $m$ is the dialogue move performed by $u$ and $r$ represents its content. Formally:

$$\begin{aligned} v(\texttt{FACTS}) &\in P^* \\ v(\texttt{QUD}_a) &\in Q^* \\ v(\texttt{QUD}_b) &\in Q^* \\ v(\texttt{PENDING}_a) &\in Utt^* \\ v(\texttt{PENDING}_b) &\in Utt^* \\ v(\texttt{LM}_a) &\in Utt^{\mathcal{D}} \\ v(\texttt{LM}_b) &\in Utt^{\mathcal{D}} \\ v(\texttt{UTT}) &\in Utt^{\mathcal{D}} \end{aligned}$$

The reason why FACTS is modelled as a stack variable is that we want to be able to check whether a particular element (i.e. some proposition) is in FACTS, and we want to be able to express this in the object language. Modelling FACTS as a variable ranging over strings of propositions allows us to use the **pop** program to check whether a particular element $x$ belongs to FACTS or not: if $x$ is in FACTS and we pop the stack repeatedly, $x$ will show up at some point as the head of the stack. Thus, we will use the notation $x \in \texttt{FACTS}$ as an abbreviation for $< \texttt{FACTS}.\mathbf{pop}^* > \mathbf{head}(\texttt{FACTS}) = x$.

[3] Arguably, there are reasons to postulate some kind of order within the set of facts. See (Ginzburg, 1997) for an account of the restrictions on which contextually presupposed facts can serve as antecedents for some anaphoric elements.

## 5 Constraining the Model

Our main aim in this section is to show that the formalism outlined previously can be used to express the rules underlying cooperative conversational interaction in terms of update operations on the DGB. The current formalisation attempts to model three different scenarios: asking and responding to a question, integrating a proposition into the commonly agreed facts, and asking for clarification when the content of an utterance has not been grounded.

In (Fernández, 2003) these scenarios were modelled in the form of complex DL programs corresponding to conventional protocols. From an abstract point of view, protocols can be thought of as a way to characterise the range of possible follow-ups in cooperative dialogue or, alternatively, as a representation of the obligations the dialogue participants are socially committed to (see (Traum and Allen, 1994; Kreutel and Matheson, 1999)). In the present paper, however, we opt for a different strategy: our aim here is to describe the appropriateness conditions for each particular scenario by means of a set of axioms, that is, a set of formulas we postulate to be valid in the model. The aim of these formulas is to restrict the operations that can be performed on the DGB components. In this sense, they can be seen as constraints characterising the appropriateness conditions of simple programs like $\texttt{UTT} := (i, \texttt{clr}, r)$ (asking a clarification question) or $\texttt{FACTS}.\mathbf{push}(x)$ (integrating an item into the common ground).

In what follows we are going to present a few of examples in detail.

### 5.1 Asking for Clarification

Following Ginzburg's account, we assume that when a dialogue participant $a$ utters an utterance $u$, $\texttt{LM}_a$ is updated with $u$. If the content of $\texttt{LM}_a$ is a question $q$, $q$ is pushed onto $\texttt{QUD}_a$. Asserting a proposition $p$ raises the question *whether p* for discussion. Thus, if the content of $\texttt{LM}_a$ is a proposition $p$, $\mathbf{whether}(p)$ will be pushed onto $\texttt{QUD}_a$. At this stage, if the addressee of $u$ can ground its

$$\forall u\ (u = (a, m, r) \wedge (\texttt{UTT} = \texttt{LM}_a = u) \wedge$$
$$((Q(r) \wedge \mathbf{head}(\texttt{QUD}_a) = r) \vee (P(r) \wedge \mathbf{head}(\texttt{QUD}_a) = \mathbf{whether}(r))) \wedge \neg G \rightarrow$$
$$<\texttt{PENDING}_b.\mathbf{push}(u)> \top \wedge \forall x\ [\texttt{PENDING}_b.\mathbf{push}(x)]\ (x = u))$$

$$\forall u\ (u = (a, m, r) \wedge (\mathbf{head}(\texttt{PENDING}_b) = \texttt{UTT} = u) \rightarrow$$
$$(\exists q\ Q(q) \wedge <\texttt{UTT} := (b, \texttt{clr}, q)> \top) \wedge$$
$$(\forall i m' q\ [\texttt{UTT} := (i, m', q)]\ (i = b) \wedge (m' = \texttt{clr}) \wedge Q(q)))$$

Table 4: Asking for Clarification

$$\forall up\ \ (u = (i, \texttt{ack}, r) \wedge (\texttt{LM}_a = \texttt{LM}_b = u) \wedge$$
$$P(p) \wedge \mathbf{head}(\texttt{QUD}_a) = \mathbf{head}(\texttt{QUD}_b) = \mathbf{whether}(p) \wedge p \notin \texttt{FACTS} \rightarrow$$
$$<\texttt{FACTS}.\mathbf{push}(p)> \top \wedge \forall x\ [\texttt{FACTS}.\mathbf{push}(x)]\ (x = p))$$

$$\forall p\ P(p) \wedge (p \in \texttt{FACTS}) \wedge (\mathbf{head}(\texttt{QUD}_a) = \mathbf{head}(\texttt{QUD}_b) = \mathbf{whether}(p)) \rightarrow$$
$$<\texttt{QUD}_a.\mathbf{pop}; \texttt{QUD}_b.\mathbf{pop}> \top$$

Table 5: Accepting a Proposition

content, she updates her LM and QUD accordingly. On the other hand, if the addressee cannot ground the content of $u$, then $u$ will be put aside and a clarification question will be posited.

Table 4 shows the axioms formalising this latter possibility. Let us have a closer look at the first formula. The antecedent describes an information state where an utterance $u$ with content $r$ is the value of UTT and $\texttt{LM}_a$, the head of $\texttt{QUD}_a$ is either $r$ (in case $r$ is a question) or $\mathbf{whether}(r)$ (in case $r$ is a proposition), and $G$ does not hold. This means that the utterance $u$ has just been posited by dialogue participant $a$ and that the addressee $b$ has not been able to ground its content. In such a situation the information state should be updated by pushing that utterance $u$ onto $\texttt{PENDING}_b$. This is expressed in the consequent of the implication, firstly by a diamond formula which guarantees that the update operation is actually being performed, and secondly by a box formula which ensures that no utterance other than $u$ can be pushed onto $\texttt{PENDING}_b$.

In the second formula, the antecedent describes a situation where an utterance $u$ with speaker $a$ is the value of both UTT and $\texttt{PENDING}_b$. That is, an utterance that has just been posited by speaker $a$ is *pending* in $b$'s information state. This situation triggers a request for clarification that should be performed by speaker $b$. This is expressed in the consequent of the formula again by means of a diamond and a box formula, which ensure that the information state will be updated by assigning to UTT an utterance $(b, \texttt{clr}, q)$ such that its speaker is dialogue participant $b$, its content is a question $q$, and the dialogue move performed is clr.

### 5.2 Proposition Acceptance

In the current formalisation, all propositions have to be acknowledged before being introduced into the commonly agreed facts. Only once an assertion has been acknowledged it is considered to be accepted by the two dialogue participants.

The axioms formalising the integration of a proposition into FACTS are shown in Table 5. The formulas follow the pattern already described in the previous subsection. In this case, the antecedent of the first formula describes a situation where an utterance $u$ performing an ack dialogue move is both the value of $\texttt{LM}_a$ and $\texttt{LM}_b$, the head's value of $\texttt{QUD}_a$ and $\texttt{QUD}_b$ is $\mathbf{whether}(p)$, where $p$ is a proposition, and $p$ is not in FACTS. This is the situation that licenses the integration of a proposition into the common ground. This is expressed by the consequent of the axiom which, again by means of a diamond and a box formula, ensures that proposition $p$ is pushed onto FACTS.

Once $p$ belongs to FACTS, $\mathbf{whether}(p)$ can be downdated from QUD. The second formula formalises precisely this situation.

$$\forall q\ (Q(q) \wedge (\mathbf{head}(\mathtt{QUD}_a) = \mathbf{head}(\mathtt{QUD}_b) = q) \wedge (\neg\exists p\ (P(p) \wedge (p \in \mathtt{FACTS})) \wedge \mathbf{ans}(p,q)) \rightarrow$$
$$\exists imr\ (<\mathtt{UTT} := (i,m,r)> \top) \wedge$$
$$\forall imr\ ([\mathtt{UTT} := (i,m,r)]\ ((m = \mathtt{ass}) \wedge P(r) \wedge \mathbf{ans}(r,q) \wedge (r \notin \mathtt{FACTS})) \vee$$
$$((m = \mathtt{ask}) \wedge Q(r) \wedge \mathbf{infl}(r,q))))$$

$$\forall pq\ P(p) \wedge Q(q) \wedge (\mathbf{head}(\mathtt{QUD}_a) = \mathbf{head}(\mathtt{QUD}_b) = q) \wedge (p \in \mathtt{FACTS}) \wedge \mathbf{ans}(p,q) \rightarrow$$
$$< \mathtt{QUD}_a.\mathbf{pop}; \mathtt{QUD}_b.\mathbf{pop} >\ \top$$

Table 6: Addressing a Question

### 5.3 Addressing a Question

Our last example concerns appropriate responses to a question under discussion. In cooperative dialogue, the optimal follow-ups after a question has been asked are either answering that question or responding with another question which influences the first one. The first formula in Table 6 formalises this observation.

The antecedent of the formula describes an information state where a question $q$ is the head's value of both $\mathtt{QUD}_a$ and $\mathtt{QUD}_b$, and $q$ has not yet been answered. The consequent of the formula expresses what the appropriate responses are in this situation. This is achieved by means of a diamond formula which guarantees that there is a state reachable by assigning some utterance $(i,m,r)$ to UTT, and a box formula which ensures that the utterance assigned to UTT will only be either an answer to the question under discussion or a question which influences it.

Once a question under discussion has been answered, it can be popped from QUD. The second formula in Table 6 formalises this situation. The antecedent of this formula has to be understood as describing an information state reached after a proposition uttered to answer a question has been acknowledged and, according to axioms in Table 5, introduced into FACTS. Once FACTS contains a proposition which is an answer to the question currently under discussion, this question can be downdated from QUD.

## 6 Discussion and Future Work

In this paper we have explored the possibility of using DL to formalise the main aspects of Ginzburg's DGB. More specifically, we have put forward a model where the components of the DGB are represented by variables ranging over different domains, while update operations are brought about by program executions that involve changes in variable assignments.

The use of DL for linguistic matters is of course not new. Several authors have observed strong parallels between the execution of computer programs and the dynamic view on discourse interpretation. The idea underlying the dynamic logic approach to the semantics of programming languages, i.e. that the meaning of a program can be captured in terms of a relation between states, has indeed been successfully applied in natural language semantics, for instance, by Groenendijk and Stokhof's *Dynamic Predicate Logic* (Groenendijk and Stokhof, 1991). Although the aims of DPL, mostly restricted to anaphorical relations across sentence boundaries, are rather different from ours, its guiding idea (i.e. that the meaning of a natural language sentence does not lie in its truth conditions, but rather in its potential to change context) is in line with the perspective taken in this paper. One could view the DGB as a semantics for utterances where each utterance is interpreted as a pair of states, i.e. as the change it brings about in the DGB.

As mention in the introduction, the current formalisation is intended as a first step towards the development of rigorous formal foundations for an approach to dialogue dynamics based on information state updates. Although this is still very much work in progress, we believe that the formalisation presented here shows that DL is an expressive and precise tool particularly well suited for this task.

From a more general point of view, we are interested in the interaction patterns that characterise different types of dialogue. In this respect, a formalisation along the same lines as the

one outlined in the present paper has been used in (Fernández, 2003) to characterise the internal structure of Inquiry-Oriented Dialogues.

There are many issues that remain still open, perhaps the most straightforward being how to use the current formalisation for instance to prove desirable properties of particular dialogue systems. In fact, some resemblances can be found between the axioms presented in Section 5 and the update rules described in (Ljunglöf, 2000), where the author presents a calculus for reasoning mathematically about the rule-based engines developed within the TRINDI project. We expect to show in our future research that some version of DL can also be successfully used to provide precise specifications of dialogue systems based on information state approaches.

## References


P. Bohlin, R. Cooper, E. Engdhal, and S. Larssson. 1999. Information states and dialogue move engines. In *IJCAI-99 Workshop on Knowledge and Reasoning in Practical Dialogue Systems.*

L. Carlson. 1983. *Dialogue Games.* Synthese Language Library. D. Reidel.

P. Cohen and H. Levesque. 1990. Rational interaction as the basis for communication. In P. Cohen, J. Morgana, and M. Pollack, editors, *Intentions in Communication.* MIT Press.

R. Cooper, S. Larsson, J. Hieronymus, S. Ericsson, E.Engdahl, and P. Ljunglof. 2001. Godis and questions under discussion. In *The TRINDI Book.*

R. Fernández and J. Ginzburg. 2002. Non-Sentential Utterances: A Corpus Study. *Traitement automatique des languages*, 43(2):13–42.

R. Fernández, J. Ginzburg, H. Gregory, and S. Lappin. 2003. SHARDS: Fragment Resolution in Dialogue. In H. Bunt and R. Muskens, editors, *Computing Meaning*, volume 3. Kluwer. To appear.

R. Fernández. 2003. A Dynamic Logic Formalisation of Inquirey-Oriented Dialogues. In *Proceedings of the 6th CLUK Colloquium*, pages 17–24, Edinburgh, UK.

J. Ginzburg. 1996. Interrogatives: Questions, facts, and dialogue. In S. Lappin, editor, *Handbook of Contemporary Semantic Theory.* Blackwell, Oxford.

J. Ginzburg. 1997. Structural mismatch in dialogue. In *Proceedings of MunDial 97.* Universitaet Muenchen.

J. Ginzburg. 1999. Ellipsis resolution with syntactic presuppositions. In H. Bunt and R. Muskens, editors, *Computing Meaning: Current Issues in Computational Semantics.* Kluwer.

J. Ginzburg. ms. A semantics for interaction in dialogue. Forthcoming for CSLI Publications. Draft chapters available from: http://www.dcs.kcl.ac.uk/staff/ginzburg.

R. Goldblatt. 1992. *Logics of Time and Computation.* Lecture Notes. CSLI Publications.

J. Groenendijk and M. Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy*, 14(1):39–100.

B. Grosz and C. Sidner. 1990. Plans for discourse. In P. Cohen, J. Morgana, and M. Pollack, editors, *Intentions in Communication.* MIT Press.

C. L. Hamblin. 1970. *Fallacies.* Methuen, London.

D. Harel, D. Kozen, and J. Tiuryn. 2000. *Dynamic Logic.* Foundations of Computing Series. The MIT Press.

J. Kreutel and C. Matheson. 1999. Modelling questions and assertions in dialogue using obligations. In *Proceedings of Amstelog 99, the 3rd Workshop on the Sematics and Pragmatics of Dialogue*, Amsterdam.

S. Larsson. 2002. *Issue based Dialogue Management.* Ph.D. thesis, Gothenburg University.

D. Lewis. 1979. Score keeping in a language game. *Journal of Philosophical Logic*, 8:339–359.

P. Ljunglöf. 2000. Formalizing the dialogue move engine. In *Proceedings of the Götalog Workshop.*

M. D. Sadek. 1991. Dialogue acts as rational plans. In *Proceedings of the ESCA/ETR workshop on multimodal dialogue.*

R. Stalnaker. 1979. Assertion. *Syntax and Semantics*, 9. Academic Press.

D. Traum and J. Allen. 1994. Discourse obligations in dialogue processing. In *Proceedings of the 32nd annual meeting of the ACL.*

D. Traum, J. Bos, R. Cooper, S. Larsson, I. Lewin, C. Matheson, and M. Poesio. 1999. A model of dialogue moves and information state revision. In *The TRINDI Book.*