

Belief Modelling for Discourse Plans

M. Garagnani[†]

University of Durham,
Durham, DH1 3LE (UK)

1. Introduction

The process of communication between agents is originated by the need of exchanging concepts. In natural language dialogues, the speaker tries to express his/her internal concepts into words and phrases; the hearer's task consists of understanding the message, i.e. translating back from natural language into his/her internal representation ('language of thought') [1].

This description underlines an important fact: when generating natural language expressions, an agent supposes that the addressee is able to understand correctly the utterances. In other words, the speaker makes *hypothesis* about the hearer *beliefs*. In fact, (s)he is assuming that the hearer believes that the terms used have a specific 'meaning' associated, and, most importantly, (s)he assumes to know which are the specific meanings held in the hearer's mind. This suggests that the process of natural language generation requires an agent to hypothesize a 'model' of, at least, the hearer beliefs and knowledge.

Although communication may be considered as an exchange of concepts (expressed using a shared language), the original motivations leading to it might go far beyond that. For example, the speaker might be intending to induce into the hearer a specific 'inference', in order to change one of his/her beliefs.¹

The theories of discourse generation have pointed out that these 'communicative goals' or intentions - which may or may not be expressed into the discourse itself - should be used to determine the *content* and the *structure* of the message [2]. In order to be able to reason about these intentions, an agent will have to maintain an adequate representation of these goals and of all the associated mental states, which requires to model not only the hearer beliefs, but also the *effects* (inference) that the speaker's utterances (speech acts) will have on those beliefs.

The decompositional nature of intentions and goals, along with the top-down refinement structure of discourse, has led to the adoption of the planning technique as a widespread approach to the problem of automatic discourse generation [11][12][13][14][16][17]. In this context, many researchers (e.g. [12] [16] [18] [19] [20] [21]) have recognised the necessity of maintaining a hearer model, and the importance of the role it plays in the planning process. Quoting Grosz and Sidner [16]:

"any model (or theory) of the communication situation must distinguish among beliefs and intentions of different agents".

Nevertheless, only few authors considered the problem of coherence and completeness of the model after the introduction of multi-nested speaker-hearer beliefs and uncertainty into the system. On the other hand, whilst research in modal logics has produced many interesting theoretical results on the subject of belief systems (e.g. [24][15]), the applications of these theories in other domains (like discourse planning) and the issues relating to their implementation, such as belief grounding, have been, in a sense, neglected.

In the first part of this paper, I shall introduce and formalise a model of beliefs for persuasive discourse generation which allows mutual beliefs, grounding of beliefs and uncertainty. In the second part, I shall illustrate how my model can be integrated into a general planning system for the generation of persuasive discourse plans.

[†] E-mail: Massimiliano.Garagnani@durham.ac.uk.

¹ In the tradition of the speech acts [3], the inference induced into the addressee is classified as 'perlocutionary act', and considered part of the communication process, instead of one of its effects.

2. The model of beliefs

This section is divided into two parts. The first contains a description of the basic characteristics of the belief model, and explains the pragmatical/philosophical reasons which have led to their implementation. In the second, a formal definition of the two languages used for the belief and nested belief expressions is presented, along with an explanation of their meaning and interaction. Finally, a description of the algorithm which I developed to realise the model of beliefs is given.

2.1. Basic characteristics

2.1.1 Framework of the belief system

The model of beliefs has been designed to be a basic component of a discourse planning system which generates plans for persuasive discourses. Hence, the structure and the characteristics of the belief system are *discourse-generation oriented*: they try to capture the important information required to construct a persuasive argument, and make them quickly available and updatable.

During the process of planning a discourse, the set of beliefs held by the system represents the current *mental state* of the speaker, and is treated by the planner as the normal *state* containing the current description of the world. The aim of the planner is to find a successful plan, i.e. a sequence of actions (*operators*) that transform the initial state (set of beliefs) into a new mental state, in which the specific (communicative) goal assigned is achieved.

Each operator takes part in the transformation *adding* and *deleting* beliefs from the set as a direct consequence of the specific ‘speech act’ that it performs. For example, if the speaker asserts a proposition *p*, the mutual belief “Speaker and hearer believe (Speaker said *p*)” will have to be added to the state, and its negation (if previously held) deleted.

The most common goals (communicative intentions) which drive a process of generation of persuasive discourses consist of *convincing* (or *dis-convincing*) the hearer about the validity of an argument (belief). Therefore, the beliefs representation will have to be sophisticated enough to allow the definition of a set of operators able to achieve this kind of ‘perlocutionary’ effect.

2.1.2 Event based knowledge

The beliefs representation of the model has, as a fundamental unit, the concept of *event*. Intuitively, an event is a concept that can be held in our mind, such as “The sky is blue”, “Matt loves Alice” or “All men are mortal”. The constituent elements of an event are basically three: *subject*, *predicate* (action) and *object*. If necessary, further features can be added, such as the time or spatial location of the event, or the destination of its object.

The conceptual unit of event is commonly used in semantic networks to express *sentences*, and represents the equivalent of a proposition in predicate logic. For example, the sentence “Bill eats the apple” assumes the two forms shown in Fig. 1, respectively as a predicate of the type *Pred(subject, object)* (a) or as a semantic network event (b).

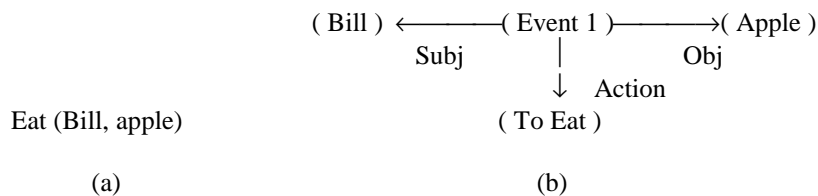


Figure 1 Predicate logic (a) and semantic network (b).

In particular, the form of event representation of Fig. 1.b constitutes the basis for the semantic network on which the natural language processing system LOLITA depends [9]. In my model of beliefs (and also in the LOLITA system), the event formalism is recursive, hence events can themselves be subject/object of other events. For example, if E1 is an event, ‘Believes(Agent,E1)’ is the event equivalent to the proposition “Agent believes that E1”.

I considered the *knowledge* of an agent as a collection of events which evolves through time. Each event belonging to this set constitutes a *belief* currently held by the agent. Therefore, the speaker's model of beliefs will consist of a set of events.

2.1.3 Support and grounding of beliefs

In logic, given a proposition p and a formal theory, if p is true (i.e. it is a theorem of that theory) then p must be either a consequence of inference rules and other propositions which are true, or one of the axioms of the theory. Similarly, in a rational belief system, the presence of a belief must be *justified*: if an agent believes E , then either E is a consequence of his reasoning about other beliefs, or it has been accepted as *grounded* belief, i.e. a belief which is not questionable [23]. The presence of a model for belief justification is also necessary to generate *persuasive* discourses. As Cohen and Perrault pointed out in [19],

"[...] before AGT will be convinced, she needs to know the justifications for AGT1's belief, which may require that AGT believes (or be CONVINCED of) the justifications for believing those justifications, etc." (p.193)²

If we adopt a 'cynical' point of view, the beliefs which we do not normally question are those deriving from our direct experience of the physical reality, i.e. our *perceptions* of the external world. For example, if I experience the event $E =$ "I see a table", then I shall strongly believe E . The importance of the experience as a way of grounding beliefs has been also underlined by various philosophers, such as Kant, Spinoza, Hume, Locke and the empiricists in general (see [10] for an interesting account on them, or [22] for a more specific example). Therefore, in my model of beliefs, I included explicit notations to represent the two mentioned aspects - reasoning and experience - of belief justification.

If a belief in E is a consequence of reasoning, then there must be another event $E1$ in the beliefs set such that the belief in $E1$ *supports* belief in E . This relationship is formally written as

SUPPORT($E1,E$)

This means that the belief in $E1$ is a *sufficient* reason to believe E , but it does not imply that either $E1$ or E have to be believed. It is important to notice that SUPPORT($E1,E$) (abbreviated with SUP($E1,E$) from now onwards) is still an event, and therefore its belief must be justified, for example using a further support event, which in turn needs a justification, and so on. It should be clear now that the belief-grounding through experience has been introduced to terminate this potentially infinite recursive process.

If the agent A believes E because (s)he has experienced it (i.e. (s)he was involved in E) then the event E does not need further support to be believed, and we shall formally write

REAL(A,E)

or, equivalently, $A_REAL(E)$. This indicates that E is a *real experience* for the agent A . Even in this case, $A_REAL(E)$ is again an event. Although the support relationship has also been introduced by other authors in the field of discourse generation (e.g. [5], who defined a symmetrical 'attack' relationship, or [2]), the idea of *experience* as a grounding of belief seems to have no precedents in this context.

2.1.4 Uncertainty of beliefs

The system of beliefs that I defined allows the inclusion of uncertain beliefs within the model. From a practical point of view, the presence of uncertainty is a fundamental aspect of the real world. To quote Clark: "Uncertainty is present in most tasks that require intelligent behaviour." [4].

In philosophy, at least since Descartes' famous 'Method of Doubt', the existence of an objective truth was questioned [10]. The limited and subjectivity-prone resources of a human being require an agent to be able to deal with uncertain information in most of the cases.

My model adopts a qualitative tripartite *degrees-of-beliefs* approach. I devised a simple model in which every event must be in *one and only one* of the following categories: *Unbelieves*, *Undecided*, *Believes*³. These degrees are used as predicates (or *modal operators*, following Hintikka's terminology) to build new events.

² In this context, AGT and AGT1 correspond respectively to hearer and speaker.

³ LOLITA's plausible inference engine works with a further refinement of these levels of certainty: {none, veryLow, low, medium, high, veryHigh, total}.

For example, if there are not enough information for the agent A to decide whether the event E should be considered true or false, then the event $Undecided(A,E)$ will be one of the beliefs present in the knowledge. Moreover, I assumed the two following axioms to hold:

$$\begin{aligned} \text{Ax1)} \quad & \text{Unbelieves}(A, E) \Leftrightarrow \text{Believes}(A, \text{NOT}(E)) \\ \text{Ax2)} \quad & \text{Undecided}(A, E) \Leftrightarrow \text{Undecided}(A, \text{NOT}(E)) \end{aligned}$$

Hence, the exhaustive and mutually exclusive opinions which might be held towards a generic event E are three: $\text{Believes}(A,E)$, $\text{Believes}(A,\text{NOT}(E))$ and $\text{Undecided}(A,E)$ (abbreviated with $A_BEL(E)$, $A_BEL(\text{NOT}(E))$ and $A_UND(E)$ from now on). The decision about which category an event E will belong to should be taken assessing the support relationship which hold in favour of E and in favour of $\text{NOT}(E)$. For example, if both E and $\text{NOT}(E)$ have valid supports (i.e. there exists $E1$ in the set of beliefs such that $A_BEL(E1)$ and $A_BEL(\text{SUP}(E1,E))$, and similarly for $\text{NOT}(E)$), then the belief $A_UND(E)$ should be adopted. This is basically the reason why the support relationship cannot be treated exactly as the logical ‘implication’: the *Modus Ponens* rule

$$A_BEL(E1), A_BEL(\text{SUP}(E1,E)) \models A_BEL(E)$$

is applicable if and only if there are no supports in favour of $\text{NOT}(E)$. For a similar analysis of the degrees of beliefs and supporting arguments, see [2].

2.1.5 Speaker’s and Hearer’s models

The belief system described so far has to be intended as the speaker’s belief model. This implies that for every event E present in the set of beliefs, also the belief $S_BEL(E)$ must hold, where ‘ S ’ means ‘Speaker’. Paragraph 2.2.2 explains how, using a simple algorithm, this potentially infinite set of beliefs can be implemented. Furthermore, in order to separate the hearer model from the rest of events which are supposed to be *unknown* by the hearer, I introduced the modal operator

$$H_UNK(E)$$

where ‘ H ’ stands for ‘Hearer’. This expression means that (the speaker believes that) the hearer has never even ‘thought’ about the event E , or about one of E ’s components. This situation might occur when the speaker believes to have some knowledge which the hearer can not possibly hold. The symmetrical form, $S_UNK(E)$ (literally, “Speaker unknowns E ”), can be present *only* as one of the hearer’s beliefs, i.e. $H_BEL(S_UNK(E))$. In fact, the expression $A_BEL(A_UNK(E))$ is contradictory for any agent A and for any event E : as long as the agent A simply ‘thinks’ about the concept $A_UNK(E)$, this becomes immediately false by definition.

2.2 The language of the beliefs

2.2.1 Formalisation of the language

The syntax for the well-formed belief expressions of the system is defined by the BNF production rules of Figure 2, where the initial symbol is $\langle WFE \rangle$ (‘Well Formed Expression’). The main characteristic of this syntax consists of the fact that it allows only two levels of adjacent nested beliefs, like in the legal expression $H_BEL(S_UND(E1))$. Each one of the two levels can be negated, i.e. can have a NOT in front of the predicate. Since every belief can be thought as having S_BEL in front of it, the number of levels is actually three. This characteristic is imposed by the two production rules for $\langle P1 \rangle$ (one level predicates) and $\langle P2 \rangle$ (two levels predicate).

$$\begin{aligned} \langle WFE \rangle & ::= \langle P1 \rangle \mid \langle P2 \rangle \mid \langle A \rangle \\ \langle P1 \rangle & ::= \langle PRED \rangle (\langle A \rangle) \mid \text{NOT} (\langle PRED \rangle (\langle A \rangle)) \\ \langle P2 \rangle & ::= \langle PRED \rangle (\langle P1 \rangle) \mid \text{NOT} (\langle PRED \rangle (\langle P1 \rangle)) \\ \langle PRED \rangle & ::= S_BEL \mid S_UND \mid S_UNK \mid H_BEL \mid H_UND \mid H_UNK \\ \langle A \rangle & ::= \langle E \rangle \mid \text{NOT} (\langle E \rangle) \\ \langle E \rangle & ::= S_REAL (\langle Ei \rangle) \mid H_REAL (\langle Ei \rangle) \mid \text{SUP} (\langle WFE \rangle, \langle WFE \rangle) \mid \langle Ei \rangle \\ \langle Ei \rangle & ::= E0 \mid E1 \mid E2 \mid E3 \mid \dots \end{aligned}$$

Figure 2 BNF formalisation of the language.

Let us dwell for a moment upon this aspect of the syntax. How many levels of belief nesting does the speaker need in order to be able to build a persuasive discourse? First of all, (s)he must be able to deal with disagreement, i.e. with expressions like $S_BEL(H_BEL(E))$, in which the hearer's (expected) belief may be different from his/her (e.g. $S_BEL(NOT(E))$). Secondly, in many examples of arguments, the disagreement among the agents would be solved if the basic 'misunderstanding' between them were simply discovered. This misunderstanding usually consists of an agent believing another agent to believe something, whereas this is not the case. In order for the speaker to be able to deal with and solve such a situation, it is necessary to include in his/her beliefs expressions like $H_BEL(S_BEL())$, in which the speaker assumes to know the hearer's model of his/her beliefs.

These two kinds of expressions have been included in the <WFE> syntax. A further level of nesting would lead to beliefs such as $H_BEL(S_BEL(H_BEL()))$, which represents (in the speaker's mind) the hearer's view of the speaker's model of the hearer's beliefs. This kind of 'reasoning', apart from being quite hard to follow and unusual in a normal situation, requires also a notable amount of knowledge and assumptions about the hearer's mental state, which is likely to be collected only in specific contexts, such as protracted dialogues or prolonged shared experiences. Furthermore, a three-level expression is commonly associated with an attempt to produce (or resist) *deception*: quoting Cohen and Perrault,

"If AGT1 successfully lied to AGT2, he would have to be able to believe some proposition p , while believing that AGT2 believes that AGT1 believes p is false (i.e., AGT1 BELIEVE AGT2 BELIEVE AGT1 BELIEVE ($\sim p$))." ([19], p.183)

My model is meant to produce persuasive discourses without using deception techniques. Therefore, as a first approximation, I restricted the syntax to only two levels of adjacent modality nesting. Nevertheless, it should be noted that expressions like $H_BEL(S_BEL(H_REAL(E1)))$ are perfectly acceptable in <WFE>.

The inner content of a predicate is defined by the production for the symbol <A>, or 'atom', which can be either an 'event' <E> or its negation. The reader should bear in mind that the distinction made here between predicates, atoms and events is due to syntactical reasons; in fact, the system will treat all of these expressions simply as events of the knowledge.

Some interesting aspects of the syntax lie in the limitations imposed to the inner arguments of a real experience $S_REAL()$ and $H_REAL()$. First of all, it is not possible to have a *negative* real experience, like $S_REAL(NOT(E0))$. The presence of this kind of expressions in the natural language is actually originated by a synthesis of a collection of positive facts. For example, the assertion "My watch does NOT work" is the result of the collection of the two repeated real experiences "Its fingers are immobile" and "It is completely silent", experienced 'continuously' during the last minute. Second, a *belief* event is not considered a real experience. If a belief were a real experience, then it would not need further support, whereas we expect the model to have a justification for every belief. In fact, a real experience event can only be an event involving the agent's physical perception, like 'to hear', 'to see', 'to feel', etc. These events will be contained into the unlimited list of 'primitive' events $E0, E1, E2, \dots$.

2.2.2 Equivalent expressions and redundant information.

The assumption of the axiom Ax2), in paragraph 2.1.4, means that two expressions like $S_UND(E)$ and $S_UND(NOT(E))$ must be considered equivalent. If we compare the expression $S_UND(H_BEL(E))$ with $S_UND(H_BEL(NOT(E)))$, we discover that they both mean that the speaker is undecided about which is the hearer's opinion about E, as Ax1) & Ax2) apply also to the hearer's model. So, they are equivalent, too.

In fact, a careful analysis of the language defined in the previous paragraph reveals that it contains many expressions which have the same meaning, and can be considered equivalent. In other words, the syntax <WFE> is semantically 'redundant', i.e. the same information can be expressed with less propositions.

It has been already mentioned that, within the planning framework, the beliefs set constitutes the state used by the planner to build a discourse plan. In order to establish the applicability of an operator in a situation, or to verify the achievement of a goal, the planner will have to check very often whether a specific belief b is contained or not in the set, i.e. whether b is true or false with respect to the current (mental) state considered. Paragraph 3.1 will explain this in further details.

If the expression b , for which the check is needed, has an equivalent form b' (like in the examples shown before), an effective check for its truth (or for the truth of b') will require to verify if either b or b' is present in the set. This 'double' check could be avoided if both of the equivalent expressions were permanently kept in the set (and both were

deleted when deleting one of them). Nevertheless, although this latter approach avoids the waste of time required by the former, it makes a bad use of the memory resources.

The alternative consists of adopting one of the forms as the ‘canonical’, and reconduct to it all the the occurrences of the other. For example, being (a) S_UND(E) equivalent to (b) S_UND(NOT(E)), we can assume (a) as the canonical form, and transform all the occurrences of the string (b) into the equivalent form (a). This requires a (semantically invariant) syntactical transformation to be performed on the expression before the check. Moreover, a new formal definition for a simpler language must be specified, such that the new language is a *subset* of the original obtained removing all the non-canonical forms. This latter approach is the one I adopted.

0)	S_BEL(a)	↔	a
1)	S_UND(H_BEL(a))	↔	S_UND(H_UND(a))
2)	S_UND(NOT(a))	↔	S_UND(a)
3)	H_UND(S_BEL(a))	↔	H_UND(S_UND(a))
4)	H_UND(NOT(a))	↔	H_UND(a)
5)	H_BEL(H_UND(a))	↔	H_UND(a)
6)	H_BEL(H_BEL(a))	↔	H_BEL(a)
7)	H_UNK(H_UND(a)) , H_UNK(H_BEL(a)) , H_UNK(H_UNK(a)) ,)	} ↔ H_UNK(a)	
10)	H_UNK(S_UND(a)) , H_UNK(S_BEL(a)) , H_UNK(S_UNK(a)) ,		
13)	H_UNK(S_REAL(a)) , H_UNK(H_REAL(a)) , H_UNK(NOT(a))		

Figure 3 List of Equivalent expressions.

Figure 3 contains the list of equivalent expressions of the language; the left hand sides constitute the non-canonical forms which can be transformed into the corresponding (canonical) right hand sides before the check is performed.

The introduction of a *pre-processing* phase, consisting of syntactical transformations of redundant expressions, leads immediately to a necessary distinction between the language previously defined (<WFE>) and the (simpler) *inner* language obtained removing all the non-canonical forms from <WFE>: this simplified, unequivocal inner language should be used to specify the expressions contained in the set of beliefs, while the *outer* <WFE> should be adopted as the language to ‘query’ the system.

The idea of using a pre-processing *algorithm* to transform an outer language expression into the equivalent inner one in order to match it against the set of (inner language) beliefs can be furthermore developed. Intuitively, the extension of the capabilities of this algorithm will reduce even more the set of expressions needed by the inner language, producing a very simple inner syntax with the same power of expressivity of <WFE>. In addition, this separation between inner and outer language, corresponding to a distinction between *explicit* and *implicit* information, and the adoption of an algorithm to reconduce any outer expression to its inner meaning, can be generally applied to any formalized language for knowledge representation.

In the following paragraphs, I shall define the simplified inner language adopted to specify the set of inner beliefs, and describe the pre-processing algorithm which evaluates the truth of any expression of the <WFE> syntax using this set of beliefs.

2.2.3 The inner language.

The outer language which I adopted is defined by the <WFE> syntax, already introduced in paragraph 2.2.1, in which only the production SUP(<WFE>,<WFE>) should be replaced with SUP(<WFF>,<WFF>). The definition of the syntax for the inner language <WFF> ⊂ <WFE> is shown below.

<WFF>	::= <A> <S_UND> <H_UND> <H_UNK> <H_BEL>
<S_UND>	::= S_UND (H_UND (<EH>))
<H_UNK>	::= H_UNK (<Ei>)
<H_UND>	::= H_UND (<EH>) H_UND (S_UND (<ES>))
<H_BEL>	::= H_BEL (<A>) H_BEL (S_UND (<ES>)) H_BEL (S_BEL (<A>)) H_BEL (S_UNK (<Ei>))
<ES>	::= H_REAL(<Ei>) SUP (<WFF>,<WFF>) <Ei>
<EH>	::= S_REAL(<Ei>) SUP (<WFF>,<WFF>) <Ei>

The production rules for $\langle A \rangle$ and $\langle E_i \rangle$ are those specified in $\langle WFE \rangle$. It should be noted that none of the redundant expressions of the outer language $\langle WFE \rangle$ can be generated in $\langle WFF \rangle$. Furthermore, this inner language does not need to contain expressions like $S_UND(E)$, where E is a simple event generated by $\langle E \rangle$. This simplification is made possible by the introduction of a *default* belief mechanism. The pre-processing algorithm calculates the truth of an expression like $S_UND(E)$ using the following definition:

$$(2.1) \quad S_UND(E) \Leftrightarrow \text{not} (E \vee \text{NOT}(E))$$

In words, if neither of the two events E nor $\text{NOT}(E)$ is present in the set of beliefs, then the belief $S_UND(E)$ should be considered true. This belief represents the *default* assumed to hold in absence of more specific information.⁴

Mutually Exclusive Expressions	Default
$S1(E) = \{E, \text{NOT}(E)\}$	$S_UND(E)$
$S2(E) = \{ H_BEL(E), H_UNK(E), S_UND(H_UND(E)), H_BEL(\text{NOT}(E)), H_UND(E) \}$	$S_UND(H_UNK(E))$
$S3(E) = \{ H_BEL(S_BEL(E)), H_UNK(E), H_BEL(S_BEL(\text{NOT}(E))), H_BEL(S_UND(E)), H_BEL(S_UNK(E)), H_UND(S_UND(E)) \}$	$H_UND(S_UNK(E))$

Table 1 Mutually exclusive expressions and associated default.

Table 1. shows the three complete sets of expressions of the language $\langle WFF \rangle$ with the associated default. Two expressions from the same set cannot hold simultaneously, i.e. each set constitutes a group of *mutually exclusive* beliefs.

2.2.4 The *OLI* algorithm

The following steps outline a simplified version of the pre-processing algorithm which has been implemented to calculate the truth value of any expression $p \in \langle WFE \rangle$ using a set of (inner) beliefs S :

1. let S' = the *extended* set of beliefs, containing S and the logical *consequences* of S ;
3. Transform p into the corresponding inner form p' , using the list of equivalent expressions .
4. If p' is a *default* expression, reconduce its proof to the check of the possible alternatives; otherwise, return True/False depending on whether p' belongs or not to the set S' .

This algorithm will be referred, from now on, as the *Outer-Language Interpreter (OLI)*. The value returned by the *OLI* algorithm is a function $OLI(p,S)$ of p and S , such that

$$OLI : Lo \times 2^{Li} \rightarrow \{\text{True}, \text{False}\},$$

where Lo and Li represent respectively the outer and inner languages (generated by $\langle WFE \rangle$ and $\langle WFF \rangle$), and 2^{Li} represents the set of subsets of Li .

⁴ In the rest of the paper, I shall refer to E and $\text{NOT}(E)$ as the beliefs *alternative* to the default.

The extended set of beliefs S' is calculated as the *closure* of S using a list of *inference rules*: an inference rule consists of an association $p/\{q,r,s,\dots\}$ among $\langle\text{WFF}\rangle$ expressions, such that if the premise p belongs to the belief set, then also the consequences $\{q,r,s,\dots\}$ should be added to the set. It should be noted that the adoption of inference rules containing only *one* proposition in the left hand side avoids the problem of *synergistic* effects during the planning process (see [6]). An extract from the list of asynergistic inference rules actually implemented in my system is shown in Figure 4.

IR1) {S_REAL(e)}	- { e }
IR2) {H_REAL(e)}	- { e, H_BEL(H_REAL(e)) }
IR3) {NOT(H_REAL(e))}	- { H_BEL(NOT(H_REAL(e))) }
IR3) {H_BEL(S_REAL(e))}	- H_BEL[cons_of S_REAL(e)] \cup { H_BEL(S_BEL(S_REAL(e))) }
IR4) {H_BEL(H_REAL(e))}	- H_BEL[cons_of H_REAL(e)]
IR5) {H_BEL(S_BEL(S_REAL(e)))}	- H_BEL(S_BEL[cons_of S_REAL(e)]) \cup { H_BEL(S_REAL(e)) }
IR6) {H_BEL(S_BEL(H_REAL(e)))}	- { H_BEL(S_BEL [cons_of H_REAL(e)]) }

Figure 4 Extract from the list of the Inference Rules; $\langle E \rangle \rightarrow e$.

An expression in the form “ $Pred [cons_of p]$ ” represents the set of beliefs $\{ Pred(c) \mid c \in Ext(p) \}$, where $Ext(p)$ constitutes the extension of $\{p\}$ obtained using (recursively) the inference rules.

However, since the right hand sides of the inference rules must contain only expression of the inner language, all the consequences $p \in Ext(p)$ such that $Pred(c) \notin \langle\text{WFF}\rangle$ should be removed.

The extension of a set of beliefs S , obtained as the *closure* of S using the list of inference rules, will contain only $\langle\text{WFF}\rangle$ expressions, and will be used by the *OLI* as current set of beliefs for the truth-evaluation of any outer expression.

3. Integration of the model in a planning system

This section describes the integration of the general model for belief systems introduced in the previous section into a generic planning system.

I shall explain how the two languages, inner and outer, defined to express respectively the beliefs of the system and the ‘queries’ (propositions to be checked by the *OLI*), should be employed within a planner which represents operators with preconditions, add and delete lists, and which adopts a mechanism of *step addition* in the goal achieving procedure. In this analysis, the important role which the *OLI* plays in the system is also made explicit.

3.1 Introduction of the new features

3.1.1 Characteristics of the planner

The functioning of a planning system is determined by three fundamental components:

1. the *state*, containing the collection of propositions which describe the current state of the world;
2. the set of *operators*, representing a list of possible ‘actions’ that can be combined into a sequence (*plan*) to achieve the desired *final state*;⁵
3. the *goal-achieving* procedure, which is the algorithm that drives the process of choosing the appropriate action (not necessarily the application of an operator) in order to achieve a specific goal.

Many different variations of these components have been developed during the history of planning (for a useful account of work in planning, see for example [8]). Nonetheless, there are few basic elements which, although named differently by different authors, can be found in most of the planners currently in use.

⁵ A plan can actually apply the same operator many times: a plan should be thought as a sequence of ‘instances’ of operators (*steps*).

First of all, operators are almost universally intended to be interpreted as actions being applicable only under specific *preconditions* and producing a set of *effects* on the current state. The preconditions consist of a list of propositions which must be true when the operator is applied.

Secondly, however complex the goal-achieving procedure may be, it will have to include the possibility of the 'step addition' mechanism, a process consisting of finding an operator which contains in its effects the specific goal to be achieved, and adding it to the plan.

One of the first and most influential planners, STRIPS [7], introduced the idea of representing the effects of an operator with two lists of propositions, called 'ADD' and 'DELETE' lists, containing respectively the list of predicates to be added to and deleted from the state at the moment of the operator's application. STRIPS' goal-achieving procedure consisted basically only of the step addition technique.

I shall start the analysis of the integration of my model into a generic planning system considering a planner model which makes use of operators with preconditions, add and delete lists, and which includes the step addition mechanism in the goal-achieving procedure. The analysis will investigate the impact that the introduction of two languages and the use of a *OLI* procedure have on these basic elements and on the state representation.

3.1.2 State representation, Add & Delete syntaxes

From the considerations made in paragraph 2.2.2, it should be clear that the state containing the collection of beliefs should be expressed using the simpler and unequivocal *inner* language, rather than the redundant *outer* one. Since the add and delete lists (from now on abbreviated with a/d) of an operator contain propositions (beliefs, events) which have to be appended to and removed from the state, they will have to consist only of expression belonging to the inner syntax. Therefore, adopting my definitions of inner and outer languages, the a/d lists should contain only expression of the language generated by $\langle WFF \rangle$.

Even if the real state is a subset of the inner language, it is possible to individuate a *virtual* state which is a set of *outer* expressions. In fact, the role of the *OLI* in the beliefs model is to allow for the check of the truth of a proposition belonging to the outer language, using an (extended) set of beliefs expressed using the inner language (see paragraph 2.2.4).

Consequently, in the planning system, the *OLI* function can be used to check for the truth of any outer expression, using, as set of beliefs, the current state (which contains only inner expressions). Therefore, the *OLI* constitutes an 'abstract platform' from which it is possible to see a 'world description' consisting of all the outer expression which are evaluated 'True' by the *OLI*. More formally, if S is the set of (inner) beliefs currently contained in the (real) state, the virtual/outer state can be defined as the set $\forall s = \{ p \in Lo \mid OLI(p,S) = True \}$.

However, in the rest of this paper, any reference to the *current state* will have to be understood as to the real (inner-language) state, if not otherwise specified.

3.1.3 Preconditions syntax and step-addition mechanism

The considerations made in the previous paragraph on the real/virtual state seem to suggest that the language which could be used to express the *preconditions* is the outer one. In fact, a precondition is a proposition p that must be true in order to allow the application of the operator, and the truth value of any outer expression p can be calculated simply applying the check $OLI(p,S)$ where S is the current state. However, although this is correct in general, there are few points which should be noted.

Let's assume that the preconditions are expressed using the outer language. During the planning process, an operator Op previously introduced to achieve a goal might contain preconditions which are not true at the moment of its application. When this happens, the planner might decide not to reject the operator, but to try finding a way to realize these unsatisfied preconditions. In STRIPS and many other planners, this is realized using a sub-goaling mechanism: an unsatisfied precondition becomes a new *goal* of the plan, and must be achieved in order to be able to apply Op .

A new goal g can then be achieved in different ways, one of which consists of the 'step-addition': this technique requires to find and add to the plan a new operator which contains g in its effects. If the effects are divided into a/d lists, then g should match one of the propositions of the add list. However, if g was originally a precondition, it belongs to the *outer* language, and matching it against an add list is not correct, as add lists contain only *inner* expressions.

A new operator should be selected for step-addition only if it achieves the outer proposition needed g ; how can we know if the operator with add list A considered (containing only inner expressions) will actually make true the outer proposition (precondition) g required? The answer to this question can be found in the *OLI* algorithm.

Let us suppose S is the current state, and consider the following method to check whether an operator Op with add list A achieves the precondition/goal p :

1. add A to S obtaining the new (possible) state S' ;
2. Op achieves p iff $OLI(p, S')$ returns 'True'.

In other words, an operator achieves a proposition p iff the truth of p follows from the application of the operator to the current state. Although this procedure results to be correct, it is more convenient to devise a method of checking which does not make use of the *current state* S .

In order to do that, let's consider only the set of beliefs contained in the add list A . We can try to check for the achievement of p testing if the truth of p follows *directly* from A ; i.e. we could adopt the condition

$$(C) \quad OLI(p, A)$$

as a criterion to check whether an operator with add list A achieves an outer proposition p .

Let us consider an example. Suppose the add list A to contain the propositions $\{S_REAL(E3)\}$, and the precondition/goal to be $p = S_BEL(E3)$. The application of the OLI algorithm to prove p using the set of beliefs A will firstly extend A to $A' = \{S_REAL(E3), E3\}$, using an inference rule such as $S_REAL(E) \vdash E$. Then, it will remove the prefix "S_BEL" from p (transformation 0. of the equivalent expressions), leaving only $p' = E3$, which will successfully match one of the propositions in A' , returning 'True'. In fact, an operator adding the inner expression $S_REAL(E3)$ to the state actually makes the outer expression $S_BEL(E3)$ true through the inference rules application. Had p been equivalent to the proposition $S_UND(E3)$ - a default expression - the OLI would have returned 'False', because the algorithm would have found present in A' one of the two 'alternative' beliefs $\{E3, NOT(E3)\}$.

Hence, the application of the condition (C) seems to represent a correct method to check whether an outer proposition p is achieved (asserted) by an operator, and therefore could be used by the step-addition mechanism to verify if an unsatisfied precondition can be realized by the addition of the operator to the plan. Nevertheless, there are two specific cases in which the simple application of this check does not give the correct answer, and therefore needs to be modified.

Default expressions.

Let us suppose the proposition p for which the check is required to be a default expression, such as $S_UND(E34)$. If the extended add list A' does not contain either of the two alternatives $\{E34, NOT(E34)\}$, $OLI(p, A)$ will return 'True', suggesting that $S_UND(E34)$ does follow from the application of the operator. This result is correct if and only if the operator actually *deletes* one of the two alternatives from the state.

If, for example, the a/d lists of the operator do not even mention the event $E34$, the result returned is wrong. Therefore, in order to check for the achievement of a default expression p , we need to *replace* the simple check (C) $OLI(p, A)$ with the following condition:

$$(C1) \quad OLI(p, A) \text{ AND } \text{not} (OLI(p, D))$$

where D is the *delete* list of the operator.

In fact, a *default* proposition p is achieved by the operator Op iff it is true in the world *after* the application of Op , and it was false in the world *before*. The truth of p afterwards is determined by $OLI(p, A)$, whereas the falsehood of p before the application of Op can be revealed by the test $OLI(p, D) = \text{False}$. Indeed, if the default p was false in the state before, and the aim of Op is to make it true, then Op must delete the previously holding alternative of p , ap . The presence of ap in D causes $OLI(p, D)$ to return 'False'.

Summarizing, a default proposition p is achieved by an operator with a/d lists A and D iff the check (C1) returns 'True'.

Negative goals

Another exception to the simple check $OLI(p, A)$ arises when the precondition checked is a *negative* expression, such as $p = NOT(H_BEL(13))$. In absence of information about the hearer's opinion on the event $E13$, the *default* mechanism

defined in my model will assume $S_UND(H_UNK(E13))$ to hold (see Table 1). This implies that, in absence of any specific assertion, $H_BEL(E13)$ is considered to be false, and thus $NOT(H_BEL(E13))$ results true. Again, $OLI(p,A)$ would give a wrong answer ('True') if the event E13 is simply not involved in the operator's effects.

Although the same reasoning which has been followed for a default expression could be repeated for a negative goal, there is one case in which, unfortunately, the criterion (C1) fails.

Let us suppose that the proposition to be checked integrates both the anomalies of *default* expression and *negative* goal, such as $p = NOT(S_UND(H_UNK(E13)))$. Let us also assume that the operator simply adds the belief $H_UND(E13)$, and deletes the previously holding one, say $H_UNK(E13)$. The check $OLI(p,A)$ will correctly return 'True'. But, when we conjunct it with the check $OLI(p,D)$, since the D list does contain $H_UNK(E13)$, then the result returned by the check is 'True', and the outcome of the condition (C1) would be 'False'. This would be incorrect, as the operator actually achieves p .

A possible solution to this problem can be found analyzing the different *reasons* which make true the test $OLI(p,A)$.

In the latter example, where this check was performed for $p = NOT(S_UND(H_UNK(E13)))$, the reason for the 'True' value returned by the OLI was the presence of the proposition $H_UND(E13)$ in the (extended) add list. On the contrary, in the example mentioned at the beginning of this paragraph, the reasons for the truth of $NOT(H_BEL(E13))$ consisted of the *absence*, in A , of any specific assertion about the hearer's opinion on E13. This suggests that, when dealing with a negative proposition p , the check to verify whether a step achieves p or not should be changed into

$$(C2) \quad OLI(p,A) \text{ AND } (Reasons(p) \cap A) \neq \emptyset$$

where $Reasons(p)$ contains the set of (independent) events which cause $OLI(p,A)$ to be true. This new check (C2) basically returns 'True' iff the proposition p holds because of some proposition contained in A .

It should be noted that when p can be made true by many independent reasons, in order for the condition (C2) to be satisfied it is sufficient that *any* of them belongs to A .

To implement the check (C2) it is necessary that the OLI function returns not only the truth value of p , but also the *reasons* which justify the result returned. For example, if $A = \{H_BEL(NOT(E1))\}$, the two checks

- a) $OLI (NOT(H_BEL(E1)), A)$
- b) $OLI (NOT(H_BEL(E45)), A)$

will both return 'True', but whilst in a) the reason returned should be the proposition $H_BEL(NOT(E1)) \in A$, in b) the OLI should return $S_UND(S_UNK(E45))$, which does not belong to A and is assumed to hold by default (see default for S2 in Table 1).

In order to obtain a single final condition which reveals the achievability of default *and* negative expressions, we can replace the two checks (C1) and (C2) with a single logical disjunction

$$(C3) = (C1) \text{ OR } (C2),$$

which gives

$$(C3) \quad OLI(p,A) \text{ AND } (\text{not}(OLI(p,D)) \text{ OR } (Reasons(p) \cap A) \neq \emptyset)$$

This condition is 'True' *if and only if* a goal p , belonging to the outer language, is achievable by an operator with a/d lists A and D (which contain only inner language expressions).

It should be pointed out that this criterion is not the equivalent of a 'Modal Truth Criterion', and does not return the 'necessary' truth of a proposition in a specific 'situation' of a partially-ordered plan (see [6]). (C3) has been defined to be used by the step-addition mechanism in order to find an operator which can realize an unsatisfied precondition p , and it does that using only the information contained in the a/d lists of the operators. Nevertheless, because of this property, it can be applied also in a partial-order planning context.⁶

⁶ An extended version of this paper, containing a theoretical formalization of the model introduced, a more detailed description of the OLI algorithm and of its integration in a partial-order planning system, is available.

4. Conclusions

In the first part of the paper, I have introduced a model of beliefs for persuasive discourse generation, containing uncertainty expressions, default and mutual beliefs, and proposed a formalization of the representation of belief *justification* and *grounding* through *reasoning* and *experience*.

In the second part of the paper, I have tried to show that the general model of belief system described can be integrated within a discourse planning system. I investigated the impact that the introduction of two languages and the use of a *OLI* procedure has on the basic elements of a general planning system. This analysis has led to the definition of a general *condition* which can be used to check for the achievement of a proposition by a specific operator of a plan. The fundamental property of this condition consists of the fact that it does not make use of the *current state* content, and results applicable also in a *partial-order* context.

Acknowledgement

I wish to thank my supervisors, Dr. M. Fox and Dr. Derek P. Long, for the invaluable help that they have given me during the development of the research which has led to these results, and during the process of drawing up of this paper. Their constant advice and precious suggestions constituted a fundamental contribute for the realisation of this work.

References:

- [1] Shapiro, S.C. (1993) "Belief spaces as sets of propositions", in JETAI-5, 225-235.
 - [2] Young, R.M., Moore, J.D., Pollack, M.E. (1994) "Towards a Principled Representation of discourse Plans", in *Proceedings of the Sixteenth Annual Meeting of the Cognitive Science Society*, Atlanta, GA.
 - [3] Searle, J.R. (1969) "Speech Acts: An essay in the philosophy of language", Cambridge University Press.
 - [4] Clark, D.A. (1990) "Numerical and symbolic approaches to uncertainty management in AI", *Artificial Intelligence Review* 4, pp.109-146.
 - [5] Dung, P.M. (1995) "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games", *Artificial Intelligence*, 77:321-357.
 - [6] Chapman, D. (1987) "Planning for Conjunctive Goals", *Artificial Intelligence*, 32(3):333-377.
 - [7] Fikes, R.E., Nilsson, N.J. (1971) "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving", *Artificial Intelligence*, 2:189-208.
 - [8] McDermott, D., Hendler, J. (1995) "Planning: What it is, What it could be, An introduction to the Special Issue on Planning and Scheduling", *Artificial Intelligence*, 76:1-16.
 - [9] Smith, M.H., Garigliano, R., Morgan, R. (1994) "Generation in the LOLITA system: an Engineering Approach", in *Proceedings of the 7th International NL Generation Workshop*, Maine.
 - [10] Scruton, R. (1995) "A short history of modern philosophy", Routledge, London.
 - [11] Maybury, M.T. (1992) "Communicative acts for explanation generation", *International Journal of Man-Machine Studies*, 37(2):135-172.
 - [12] Moore, J.D., Paris, C.L. (1993) "Planning text for advisory dialogues: Capturing intentional and rhetorical information", *Computational Linguistics*, 19(4):651-695.
 - [13] Young, R.M., Moore, J.D. (1994) "DPOCL: A principled Approach to Discourse Planning", in *Proceedings of the 7th International Generation Workshop*, Kennebunkport, Maine, pp.171-180.
 - [14] Hovy, E.H. (1993) "Automated discourse generation using discourse structure relations", *Artificial Intelligence*, 63:341-385.
 - [15] Halpern, J.Y., Moses, Y. (1992) "A guide to completeness and complexity for modal logics of knowledge and belief", *Artificial Intelligence*, 54:319-379.
 - [16] Grosz, B.J., Sidner, C.L. (1990) "Plans for discourse", in Cohen P., Morgan J., Pollack M.E., eds. *Intentions in Communication* (MIT Press, Cambridge, MA), pp.417-444.
 - [17] Moore, J.D., Swartout, W.R. (1990) "Dialogue based explanation", in Paris C.L., Swartout W.R., Mann W.C., eds. *Natural Language in Artificial Intelligence and Computational Linguistics* (Kluwer, Boston, MA), pp.3-48.
 - [18] Walker, M.A., Rambow, O. (1994) "The Role of Cognitive Modeling in Achieving Communicative Intentions", in *Proceedings of the 7th International Generation Workshop*, Kennebunkport, Maine, pp.171-180
-

- [19] Cohen, P.R., Perrault, C.R. (1979) "Elements of a Plan-Based Theory of Speech Acts", *Cognitive Science*, 3:177-212.
- [20] Grosz, B.J., Sidner, C.L. (1986) "Attention, intention, and the structure of discourse", *Computational Linguistics*, 12(3):175-204.
- [21] Asher, N., Koons, R. (1993) "The revision of Beliefs and Intentions in a Changing World", in *Proceedings of the AAI Spring Symposium Series: Reasoning about Mental States: Formal Theories and Applications*.
- [22] Ackermann, R.J. (1972) "Belief and Knowledge", Anchor, New York.
- [23] Reed, C., Fox, M., Long, D.P., Garagnani, M. (1996) "Persuasion as a Form of Inter-Agent Negotiation", *Lecture Notes in Artificial Intelligence* 1286, pp.120-136.
- [24] Smith, R.S. (1991) "Modal logic", *Artificial Intelligence Review* 5, pp.5-34.