Logic-based Rhetorical Structuring for Natural Language Generation in Human-Computer Dialogue

Vladimir Popescu[1,2], Jean Caelen[1], and Corneliu Burileanu[2]

[1] Laboratoire d'Informatique de Grenoble, Grenoble Institute of Technology, France
{vladimir.popescu, jean.caelen}@imag.fr
[2] Faculty of Electronics, Telecommunications and Information Technology, University "Politehnica" of Bucharest, Romania

Abstract. Rhetorical structuring is field approached mostly by research in natural language (pragmatic) interpretation. However, in natural language generation (NLG) the rhetorical structure plays an important part, in monologues and dialogues as well. Hence, several approaches in this direction exist. In most of these, the rhetorical structure is calculated and built in the framework of Rhetorical Structure Theory (RST), or Centering Theory [7], [5]. In language interpretation, a more recent formal account of rhetorical structuring has emerged, namely Segmented Discourse Representation Theory (SDRT), which alleviates some of the issues and weaknesses inherent in previous theories [1]. Research has been initiated in rhetorical structuring for NLG using SDRT, mostly concerning monologues [3]. Most of the approaches in using and / or approximating SDRT in computer implementations lean on dynamic semantics, derived from Discourse Representation Theory (DRT) in order to compute rhetorical relations [9]. Some efforts exist in approximating SDRT using less expressive (and expensive) logics, such as First Order Logic (FOL) or Dynamic Predicate Logic (DPL), but these efforts concern language interpretation [10]. This paper describes a rhetorical structuring component of a natural language generator for human-computer dialogue, using SDRT, approximated via the usage of FOL, doubled by a domain-independent discourse ontology. Thus, the paper is structured as follows: the first section situates the research in context and motivates the approach; the second section describes the discourse ontology; the third section describes the approximations done on vanilla SDRT, in order for it to be used for language generation purposes; the fourth section describes an algorithm for updating the discourse structure for a current dialogue; the fifth section provides a detailed example of rhetorical relation computation. The sixth section concludes the paper and gives pointers to future research and improvements.

1 Introduction

This paper describes a module for rhetorical structure computation, for a natural language generator in human-computer dialogue. More specifically, pragmatic aspects concerning rhetorical coherence of speech turns in dialogue are discussed. It is known that when pragmatic aspects are concerned, total domain or application independence is an elusive goal, since the "pragmatics" of a fragment of speech strongly depends on the context in which that fragment occurs.

The research described in this paper is performed in the framework of a project consisting in the development of a task-oriented dialogue system, designed in a generic manner (i.e., easily customizable to different tasks and applications) and applied to several domains, such as meeting room reservation, in an enterprise, or book reservation, in a library [2]. Our team has already designed and implemented components regarding dialogue management, task planning, semantic parsing of user's requests, and pragmatic interpretation of these. For pragmatic interpretation purposes, SDRT has been adapted and extended for dialogue, in order to integrate the concept of *topic* [2]. However, the component responsible for generating system's responses has been reduced to a template-based generator, thus providing little flexibility and robustness with respect to contextual variations, and lacking pertinence with respect to dialogue dynamics [6].

As for the use of SDRT in NLG, research of L. Danlos or L. Roussarie could be put forward [3]; for the extensions performed on SDRT in order to integrate aspects related to dialogue (however, for interpretation purposes), work of L. Prevot and his team could be mentioned [4]. For approximations or reformulations of SDRT using less "heavy" logics, work of M. Staudacher [10] could be mentioned, for example.

The novelty of the approach described in this paper resides in the usage of a logic formalism less complex (from a computational point of view) than dynamic semantics or logics - first order predicate logic, parameterized by a discourse ontology specifying the scopes of the entities invoked in logic formulas. The advantage of this approach is, at a practical level, that it allows the usage of software tools and environments designed for FOL (for instance, several flavors of PROLOG) in a straightforward manner. Then, from a methodological point of view, the approach proposed is task-independent, since the discourse ontology does not depend on the constraints imposed by a specific application domain; the coupling with the task-specific aspects is made via a task ontology, handled by the task controller in the dialogue system [2], [6]. This allows for an augmented portability of the generation module, thus lowering the costs for the adaptation to a new task.

The following section describes the particular elements chosen for the formal expression of logic formulas in a discourse ontology (that is generic, i.e, task-independent); the third section presents the approximation of a fragment of SDRT using FOL and integrating a set of semantics for the rhetorical relations being used in dialogues; the fourth section describes an algorithm for updating the discourse structure (called Segmented Discourse Structure - SDRS) of a current dialogue; the fifth section provides a detailed example of rhetorical relation computation in the SDRS update process. Finally, the sixth section concludes the paper and provides pointers to further work.

2 Task-Independent Discourse Ontology

A logic formula expressed in FOL contains five types of information: (i) connectives: $\wedge, \vee, \neg, \Rightarrow$, (ii) quantifiers: $\forall, \exists$, (iii) objects, and (iv) predicates.

Type (i) entities can link different logic formulas; type (ii) entities can precede type (iii) entities that, in their turn, can be followed by type (iv) entities. Type (iv) entities can be preceded by type (iii) entities.

In order to be able to state in detail the content of the logic formulas expressing utterances, facts and rules in the ontology (described hereafter), one takes into account that the dialogues concerned involve negotiation on time intervals. Thus, a taxonomy of possible *moments of time* is defined, in order to augment the expressive power of FOL. These temporal markers are: (i) $t_{\#}$ - present conditional, (ii) t_{+} - future and "new", (iii) t - present, (iv) t_{-} - past simple and "old", (v) $t_{=}$ - past perfect, (vi) $t_{\pm}$ - past conditional, (vii) $t_{\mp}$ - future in the past, (viii) $t_{\exists}$ - a unique moment not precisely situated on the time axis, (ix) $t_{\forall}$ - any moment, eternal, permanent.

As for the discourse ontology for "pragmatic" generation (a.k.a. rhetorical structuring), its elements are represented by entities in a knowledge base. This knowledge base specifies a set of objects, functions and predicates involved in the expression of the *semantics* of the rhetorical relations used, in the framework of SDRT, and is used by the generator to infer the rhetorical relations between utterances[3].

For this knowledge base, a minimal set of predicates is defined: (i) $\in$ - MemberOf, (ii) $\subset$ - SubclassOf, (iii) $\ni$ - ClassOf, (iv) $\supset$ - SuperclassOf, (v) $\cap = \emptyset$ - Disjoint, (vi) $\cup =$ All - ExhaustiveDecomposition, (vii) $(\cap = \emptyset) \wedge (\cup = \mathsf{All})$ - Partition. In order to express **measures**, one defines the predicates: smaller, greater, and equals. For temporal event handling, a specific set of predicates could be defined, as in [8], but in our generator these latter ones are not used, therefore are not mentioned here.

In order to enforce the structure of the ontology so that it remains generic with respect to the task and at the same time flexible and useful, the ontology contains the following particular entities: (i)functions: $SARG()$, $Plan()$; (ii) predicates: answer(), bad_time(), good_time(), emitter(), receiver(), topic(), enounce(), question().

The discourse ontology contains elements (terminals, i.e, without subtypes) whose properties must be specified via a set on axioms, denoting by Ω the set of values for the object in the task ontology[4], by $K(\alpha)$ the clause logically expressing the semantics of the utterance α and by t_{α}, the moment in time when utterance α is produced. Hence, the specific predicates and functions are: equals(α, question), equals(α, enounce), equals(β, confirmation(α)), equals(β, answer(α)), topic(α), emitter(α), entails(α, β), $SARG(\alpha)$, good_time(Δt_{β}), bad_time(Δt_{β}), and Δt_{β}. For example, the axiom specifying the semantics of topic(α)is given below:

topic(α) ::= ExhaustiveDecomposition($i, j; v_i, \omega_j$) $\wedge$ MemberOf($v_i, K(\alpha)$)$\wedge$

MemberOf(ω_j, Ω) $\wedge$ ($\exists k$: equals(v_k, ω_j) $\wedge$ MemberOf($v_k, K(\alpha)$)).

One notices that, unlike the classical SDRT, the notion of topic of an utterance is defined here in terms of sets of objects in the domain ontology, referred to in a determined manner[5] in the utterance. Hence, the topic relations between utterances are computed using the task/domain ontology, handled by the task controller.

---

[3] The utterances are represented as clauses in FOL.

[4] The rhetorical structuring component in the linguistic generator adapts to the task in this manner: the task-dependent aspects are provided by the task manager that handles, in its turn, a task ontology, different from the discourse ontology.

[5] Here, an object is referred to in a determined manner if and only if the logic variable designating the object in the ontology has an allowed value, assigned in the logic clause expressing the semantics of the utterance.

3 SDRT adaptation for Language Generation in Dialogue

For the goals supposed by our project, a subset of SDRT has been chosen, namely 17 rhetorical relations, specified below; these rhetorical relations are due to be used at the "pragmatic" level for rhetorical structuring of the utterances to be generated. These rhetorical relations are grouped in three categories:

– **first-order** relations - relations strongly related to *temporal* aspects in dialogue; these relations will be used in an approximative manner, specific to the characteristics of the type of dialogue concerned (see below); these relations are: Q-Elab, IQAP, P-Corr and P-Elab, with the informal semantics stated in [1];

– **second-order** relations - relations less constrained by the temporal context of the dialogues concerned; by consequence, these relations are used in the generator in the same manner as specified by vanilla SDRT [1]; these relations are: Background$_q$, Elab$_q$, Narration$_q$, QAP, ACK and NEI;

– **third-order** relations - relations specific to monologues and used to relate utterances within a speech turn, generated by one speaker, either the human (U), or the machine (M); these relations are: Alternation, Background, Consequence, Elaboration, Narration, Contrast and Parallel, with informal semantics defined in [1].

These rhetorical relations are speech act types, reflecting the dependencies between the success of the (performance of the) current speech act and the content of a set of preceding acts in dialogue. Accordingly, a dialogue is considered pragmatically *coherent* if and only if for any utterance there exists at least one connection, via a discourse relation, between it and another speech act belonging to the dialogue history (i.e., to the set of preceding speech turns in dialogue). Such a connection is called SARG ("Speech Act Related Goal").

In language pragmatic interpretation, SARGs are recovered out of the discourse context, whereas in generation, these SARGs are defined by the speech acts *enclosed* in the clauses come from the dialogue controller [2], being thus priorly known. Hence, in language generation the issue is not the identification of the SARGs, but their *placement* in an existing discourse context.

In order to enforce rhetorical coherence by taking into account pragmatic aspects in language generation, we consider only a particular type of dialogues, encountered in meeting room reservation, book reservation in a library, or, more generally, in situations where the agreement on time intervals for the usage of a specified set of resources is essential. Hence, the characteristics of such a dialogue are: (i) the limited scope of the goals, and (ii) the importance of temporal aspects.

As in work reported by Schlangen et al. [9], the SDRT is approximated in that the answers that the system is supposed to generate are restricted to information concerning (i) description of time intervals of availability of a certain book or item, in relation to the interval Δt, convenient for the user; (ii) the adequacy or non-adequacy of the interval Δt, i.e., whether it is a "good" time interval or not (in other words, whether in the interval Δt the user having made a request to the system may have access to the book he wishes for, or not).

Therefore, the generator ought to produce utterances situating the time interval proposed by the user U, Δt, in terms of appropriate time for the loan (good_time(Δt)), or not appropriate (bad_time(Δt)).

The rules for computing the discourse relations situating the communicative intention determined by the dialogue controller, with respect to discourse context are monotonic in our model, while in vanilla SDRT there are non-monotonic. This choice is motivated by the possibility to avoid thus computationally intensive consistency verifications. These verifications would have been necessary if one had used non-monotonic rules, since the truth value of the semantics inferred via these rules could have changed at each discourse structure update.

The knowledge of the prior general goal of the users in dialogue, in relation to the temporal aspects of the conversation, allows us to state the semantics of the first-order rhetorical relations. For example, the semantics of Q-Elab is specified below:

Q-Elab(α, β) ::= equals $(\alpha, \text{enounce}) \wedge \text{equals}(\beta, \text{question}) \wedge \neg\text{Disjoint}(\Delta t_\beta, SARG(\alpha))$
$\wedge\, (\forall\ \text{answer}\,(\beta))\, \neg \Rightarrow \neg SARG(\alpha)$.

In words, this formula states that the relation Q-Elab can be inferred between utterances α and β if and only if β is a question so that any answer to it elaborates a plan for satisfying the SARG of α.

The semantics of the second-order and third-order rhetorical relations are specified in a similar manner; for space considerations, their semantics are not given here.

4 Dialogue SDRS Update

This section presents an algorithm for updating the discourse structure (SDRS) of a dialogue, adding an utterance to it[6]. More specifically, for a clause $K(\alpha)$, come from the dialogue controller, expressing the communicative intention to be put in an utterance α, one wishes to find the set $\Re$ of discourse relations relating utterance α to a set $\aleph$ of previous utterances in dialogue.

The algorithm for updating the dialogue SDRS is presented below:

1. for each utterance labeled α to add to the SDRS (and generate):
   (a) read $K(\alpha)$, through a query to the dialogue controller;
   (b) perform initializations: $\aleph(\alpha) \leftarrow \emptyset$, $\Re(\alpha) \leftarrow \emptyset$;
   (c) for each utterance β_i in $\bar{\aleph}$ $(i = 1, ...|\bar{\aleph}|)$:
      i. for each rhetorical relation ρ_j known $(j = 1, ...17)$:
         A. retrieve the logic formulas $K(\beta_i)$ and Σ_j;
         B. compute the truth value of $\Sigma_j\,(\sigma\,(K(\alpha), K(\beta_i)))$ and denote it by γ;
         C. if $\gamma =$FALSE, then go to 1.(c).i.;
            else, perform $\Re \leftarrow \Re \cup \{\rho_j\}$ and $\aleph \leftarrow \aleph \cup \{\beta_i\}$ and go to 1.(c).i.;
         D. perform: $\aleph(\alpha) \leftarrow \aleph(\alpha) \cup \aleph$ and $\Re(\alpha) \leftarrow \Re(\alpha) \cup \Re$;
2. compute the truth value of equals$(|\aleph(\alpha)|, 0) \wedge$ equals$(|\Re(\alpha)|, 0)$ and denote it by υ:
   (a) if $\upsilon =$FALSE, then:
      i. add α to the utterances in the current SDRS;
      ii. add $\Re(\alpha)$ to the rhetorical relations in the current SDRS;
   (b) else build a new SDRS having one utterance, α, and no rhetorical relations.

[6] An utterance is added to a SDRS if and only if at least a rhetorical relation between it and utterances in the dialogue history is found.

In the algorithm specified above, we denoted by $\bar{\aleph}$ the set of *all* the utterances preceding α in the current dialogue and we searched, for each utterance β_i in the dialogue history, the rhetorical relations ρ so that $\Sigma_\rho\left(\sigma\left(K(\alpha), K(\beta_i)\right)\right) \neq \mathsf{FALSE}$, where σ denotes the permutation operation, Σ_ρ denotes the formal semantics of ρ, $|X|$ denotes the number of elements in the set X, and $\cup$, the set union operator.

This algorithm builds a *complete* discourse structure, in that all the logically possible rhetorical relations between pairs of utterances (out of which at least one is due to be generated by the machine) are found; the possible contradictions involved by this approach (stemming from the fact that a rhetorical relation might be revised by consequent rhetorical relations or future utterances) are alleviated by the monotonicity assumption governing the logic framework used. This leads to the creation of all the potential discourse structures, without finding the most coherent one (in the sense discussed in [1]).

5 Rhetorical Relations Computation: Extended Example

In the algorithm presented in the previous section for updating the SDRS for a dialogue, the essential step is 1.(c).i.B; this step will be detailed through an example in this section. More specifically, one will show the manner in which the truth values for the clauses $\Sigma_j\left(\sigma\left(K(\alpha), K(\beta_i)\right)\right)$ are computed, where i spans over the set of utterances, available as logic clauses in a discourse structure, and j, over the set of rhetorical relations known.

We consider the following dialogue fragment, in French language (English translations are provided in italics below each utterance):

U : α : Je lirai ce livre lundi.
(*I will read this book on Monday.*)
M : β : Est-ce bien pour vous à 14 h ?
(*Is it good for you at 2 P.M.?*)

In fact, the utterance α is come from the user, thus available at the same time as text and as clause expressing the meaning of the utterance[7], while the utterance β is available only as a clause, expressing a communicative intention, computed by the dialogue controller [2], [6]; its textual form is due to be determined by the language generator. More specifically, between utterances α and β the rhetorical relation Q-Elab can be inferred. However, the generator cannot know it in advance, thus it has to try each of the 17 rhetorical relations used; we suppose, for simplicity, that it is precisely the relation Q-Elab(α, β) that is checked. Hence, the following processing stages are performed:

1. Find the clauses expressing the semantics of the utterances α and β:
   – $\alpha \mapsto K(\alpha) ::= \exists X, Y : \mathsf{object}(X) \wedge \mathsf{equals}(X, \mathsf{book}) \wedge \mathsf{agent}(Y) \wedge \mathsf{equals}(Y, U) \wedge \mathsf{equals}(t_\alpha, t) \wedge \mathsf{read}(Y, X) \wedge \mathsf{equals}(\Delta t_\alpha, t_+)$;
   – $\beta \mapsto K(\beta) ::= \exists X, Y : \mathsf{object}(X) \wedge \mathsf{equals}(X, \mathsf{book}) \wedge \mathsf{agent}(Y) \wedge \mathsf{equals}(Y, U) \wedge \mathsf{read}(Y, X) \wedge \mathsf{greater}(t_\beta, t_\alpha) \wedge \mathsf{equals}(\Delta t_\beta, \text{'14h'}) \wedge \mathsf{equals}(\Delta t_\beta, t_+) \wedge \mathsf{equals}(\mathsf{good\_time}(\Delta t_\beta), ?)$;

[7] The clause associated to an utterance come from the user is computed by a semantic parser together with a pragmatic interpreter [6].

2. Retrieve the semantics of the rhetorical relation currently checked:
$\Sigma_{\text{Q-Elab}}$ ::= equals $(\alpha, \text{enounce}) \wedge$ equals$(\beta, \text{question}) \wedge \neg$Disjoint$(\Delta t_\beta, SARG(\alpha)) \wedge$ $(\forall$ answer $(\beta) \neg \Rightarrow \neg SARG(\alpha))$;
3. Expand each entity in the semantics of the rhetorical relation currently checked:
– equals$(\alpha, \text{enounce})$ ::= $\neg$equals$(\alpha, \text{question}) \mapsto \forall v$: MemberOf$(v, K(\alpha)) \vee \exists \omega$: MemberOf$(\omega, \Omega) \vee$ equals(v, ω);
– equals$(\beta, \text{question})$ ::= $\exists v'$: MemberOf$(v', K(\beta)) \wedge \neg \exists \omega'$: MemberOf$(\omega', \Omega) \wedge$ equals(v', ω');
– $SARG(\alpha)$ ::= $\exists X, Y, \theta$: object$(X) \wedge$ equals$(X, \text{book}) \wedge$ agent$(Y) \wedge$ equals$(Y, U) \wedge$ good_time$(\theta) \wedge$ equals$(\theta, \Delta t_\alpha) \wedge$ greater$(\theta$, 'lundi');
– Δt_β ::= '14h' $\wedge$ 'lundi';
– $\forall$answer(β) ::= $\forall \delta$: equals$(\delta$, answer$(\beta)) \mapsto \forall \delta$: greater$(t_\delta, t_\beta) \wedge$ equals(topic(δ), topic$(\beta)) \wedge \forall v''$: MemberOf$(v'', K(\delta)) \Rightarrow \exists \omega''$: MemberOf$(\omega'', \Omega) \wedge$ equals(v'', ω'');
– $\neg SARG(\alpha)$ ::= $\forall X, Y, \theta$: object$(X) \vee$ equals$(X$book$) \vee$ agent$(Y) \vee$ equals$(Y, U) \vee$ bad_time$(\theta) \vee$ equals$(\theta, \Delta t_\alpha) \vee$ greater$(\theta$, 'lundi');
– topic(β) ::= ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+);
– good_time(θ) ::= $\exists \gamma, \pi$: $\neg$Disjoint(topic(γ), topic$(\pi)) \wedge$ smaller$(t_\alpha, t_\pi) \wedge$ (SubclassOf$(\theta, \Delta t_\alpha) \vee$ equals$(\theta, \Delta t_\alpha)) \wedge \pi$: equals$(\Delta t_\pi, \theta)$;
4. Compute the truth value of each clause between conjunctions, in the semantics of the rhetorical relation currently checked:
– equals$(\alpha, \text{enounce}) \wedge$ equals$(\beta, \text{question}) \rightarrow$ TRUE; this expression is obtained substituting the semantics of α in the semantics of the clause shown above and exploring in the discourse ontology *and* in the task ontology (Ω) all the possible values for the variables involved;
– $\neg$Disjoint$(\Delta t_\beta, SARG(\alpha)) \rightarrow$ TRUE; this results directly from the clause: SubclassOf$(\Delta t_\beta, \Delta t_\alpha) \wedge$ SubclassOf$(\Delta t_\alpha, SARG(\alpha))$;
– $\forall$ answer $(\beta) \neg \Rightarrow \neg SARG(\alpha) \mapsto \forall \delta$: equals$(\delta$, answer$(\beta))$ ::= $\forall \delta$: greater$(t_\delta, t_\beta) \wedge$ equals(topic(δ), ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+)) $\wedge \forall v''$: MemberOf$(v'', K(\delta)) \Rightarrow \exists \omega''$: MemberOf$(\omega'', \Omega) \wedge$ equals$(v'', \omega'') \neg \Rightarrow \forall X, Y$: object$(X) \vee$ equals$(X, \text{book}) \vee$ agent$(Y) \vee$ equals$(Y, U) \vee$ bad_time$(\Delta t_\beta) \vee$ equals$(\Delta t_\beta, \Delta t_\alpha) \vee$ greater$(\Delta t_\beta$, 'lundi');
the last formula was obtained substituting the predicates and functions by their explicit definitions, and the variable θ, by Δt_β; then, by *skolemization* of the variables in this formula, one has:
$\forall$ answer $(\beta) \neg \Rightarrow \neg SARG(\alpha) \mapsto \forall \delta$: equals$(\delta$, answer$(\beta)) \mapsto \forall \delta$: greater$(t_\delta, t_\beta) \wedge$ equals(topic(δ), ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+)) $\wedge \forall v''$: MemberOf$(v'', K(\delta)) \Rightarrow$ MemberOf$(\omega_0, \Omega) \wedge$ equals$(v''$, $\omega_0) \neg \Rightarrow \forall X, Y$: object$(X) \vee$ equals$(X, \text{book}) \vee$ agent$(Y) \vee$ equals$(Y, U) \vee$ bad_time(Δt_β) $\vee$ equals$(\Delta t_\beta, \Delta t_\alpha) \vee$ greater$(\Delta t_\beta$, 'lundi');
then, universal quantifiers are eliminated and the possible paths in the task ontology are explored, obtaining non-contradiction, hence the value TRUE.

This example shows that it is possible, in principle, to compute a rhetorical relation connecting a current utterance (due to be generated in textual form), available only as a logic formula, to a dialogue in progress, using FOL and a task-independent discourse ontology. The computational costs of the algorithm proposed is yet to be evaluated using a dialogue corpus (acquired by the Wizard of Oz method [2], such as the PVE - "Portail Vocal pour l'Entreprise" corpus [6], [11]).

6 Conclusions and Further Work

This paper has presented a rhetorical structuring component of a natural language generator for human-computer dialogue. The pragmatic and contextual aspects are taken into account communicating with a task controller providing domain and application-dependent information, structured in a task ontology. In order to achieve the goal of computational feasibility and genericity, SDRT has been strongly adapted to natural language generation in dialogue. Thus, a discourse ontology has been defined and several axioms structuring it have been specified; moreover, specific predicates and functions have been given a formal account. Then, using this ontology, a set of semantics, in first-order predicate logic, has been specified for a fragment of SDRT. The advantage of this approach resides in the possibility to readily use software tools designed for FOL and in the relative simplicity of the formal statement; this latter point allows for straightforward extensions or customizations to different types of dialogue (e.g. tutoring dialogue). The implementation of the rhetorical structuring component (described in this paper) in ISO PROLOG is currently under way.

However, several improvements could be brought to current processing stages, mostly with respect to the computational cost involved and secondly regarding the precision and reliability of the rhetorical relation computation process. The reduction of discourse structure updating time could be achieved by limiting the dialogue history, for a current utterance, to N previous utterances; psycholinguistic studies motivate a choice of $N = 7$ [8]. The goal of augmenting the reliability of the inference could be achieved by limiting the number of candidate rhetorical relations, for a given pair of utterances; this could be done by using speech acts to characterize the utterances from an illocutionary point of view [2], [11]. Thus, for a given pair of utterances, a mapping is built between the corresponding pair of speech acts and the set of possible rhetorical relations connecting them; in this respect, a study of our team has already been done [11].

References

1. N. Asher and A. Lascarides, *Logics of Conversation*, Cambridge University Press, 2003.
2. J. Caelen and A. Xuereb, *Interaction et pragmatique*, Editions Hermès, Paris, 2007.
3. L. Danlos and D. El-Ghali, "A complete integrated NLG system using AI and NLU tools", *Proceedings of COLING'02*, Taiwan, 2002.
4. N. Maudet, Ph. Muller and L. Prevot, "Tableaux conversationnels en SDRT", *Actes de TALN 2004*, Fès, Maroc, 2004.
5. M. F. McTear, "Spoken Language Technology: Enabling the Conversational User Interface", *ACM Computing Surveys*, vol. 34, no. 1, 2002.
6. H. Nguyen, *Dialogue homme–machine : Modélisation de multisession*, PhD Thesis, Joseph Fourier University, Grenoble, France, 2005.
7. E. Reiter and R. Dale, *Building Natural Language Generation Systems*, Cambridge University Press, 2000.
8. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2003.
9. D. Schalngen et al., "Resolving Underspecification using Discourse Information", *Proceedings of BI-DIALOG 2001*, Springer, 2001.
10. M. Staudacher, *SDRT Reformulated using DPL*, Term Paper, Bielefeld University, 2005.
11. A. Xuereb and J. Caelen, "Actes de langage et relations rhétoriques en dialogue homme–machine", *Séminaire Logique et Dialogue*, France, 2004.