Structuration rhétorique en génération du langage naturel pour le dialogue homme-machine

Vladimir POPESCU[1, 2], Jean CAELEN[1], Corneliu BURILEANU[2]

[1] Laboratoire d'Informatique de Grenoble, Institut National Polytechnique de Grenoble, France
{vladimir.popescu, jean.caelen}@imag.fr
[2] Faculté d'Electronique, Télécommunications et Technologie de l'Information,
Université "Politehnica" de Bucarest, Roumanie

1 Introduction

Le but de cet article est de présenter un module de calcul de la structure rhétorique pour un système de géneration du langage naturel en dialogue homme-machine. Plus précisément, il s'agit de décrire les démarches au niveau pragmatique. On sait que lorsqu'il s'agit de traitements visant la prise en compte des aspects pragmatiques d'une *manifestation* langagière, qu'elle soit sous forme monologique ou dialogique, une indépendance totale de l'application n'est pas possible, car la "pragmatique" d'un texte ou énoncé dépend fortement du contexte où il apparaît et se manifeste.

Les recherches décrites dans cet article se situent dans le contexte d'un projet consistant à développer un système de dialogue homme-machine finalisé par une tâche précise, la réservation des salles de réunion, dans le cadre d'une entreprise ; notre équipe a déjà conçu et réalisé sous forme de démonstrateurs des composantes concernant la gestion du dialogue et de la tâche, la compréhension des requêtes provenues de l'utilisateur, et l'interprétation pragmatique de ces dernières. Pour l'interprétation pragmatique, la théorie SDRT a été adaptée et étendue pour le dialogue, afin d'intégrer la notion de topique [4]. Par contre, le composant responsable avec la génération des réponses du système face aux requêtes des utilisateurs était réduit à une approche de génération "à trous", offrant peu de flexibilité et faisant défaut de pertinence par rapport au contexte et à la dynamique du dialogue [15].

Ceci étant, cet article décrit en partie les démarches à effectuer dans un module de génération pour le dialogue homme-machine. Pour ce module, une architecture à plusieurs niveaux, à la fois compatible et portable par rapport aux architectures de référence en génération -telles que RAGS ou le modèle de Reiter et Dale-, a été proposée en [18]. En fait, seule la génération dite "pragmatique", concernant surtout la structuration rhétorique des répliques à générer et leur insertion dans le dialogue courant, fait l'objet de cet article. Pour cela, le choix de la SDRT s'inscrit de manière naturelle dans les recherches menées par notre équipe en dialogue homme-machine.

En ce qui concerne l'utilisation de la SDRT en génération, les recherches de L. Danlos, L. Roussarie [8], [9], etc pourraient être citées ; pour l'extension de la SDRT au dialogue, hormis les travaux d'A. Xuereb et J. Caelen [26], [4], on pourrait citer les travaux de L. Prévot [12]. Pour les approximations ou reformulations de la SDRT utilisant des formalismes logiques moins "lourdes" que la sémantique dynamique dérivée de la DRT ("Discourse Representation Theory"), on pourrait citer les travaux de M. Staudacher [23], par exemple.

Ce que cet article apporte c'est l'appui sur une logique moins complexe (et donc moins expressive) que les sémantiques dynamiques -la logique des prédicats du premier ordre, paramétrée par une ontologie discursive dressant les champs d'action pour les entités invoquées. L'avantage offert par cette approche est d'abord d'ordre pratique, technologique, dans le sens que l'utilisation des outils logiciels existants pour les traitements concernant la logique du premier ordre - PROLOG par exemple - devient possible. Ensuite, même du point de vue méthodologique, l'approche proposée apporte un plus de généricité, car l'ontologie discursive est indépendante de la tâche ; le couplage avec cette dernière est fait par une ontologie de la tâche, gérée par le cotrôleur de tâche. Cela entraîne une portabilité accrue de notre module, ce qui implique moins de temps nécessaire pour le porter vers une nouvelle tâche.

On conçoit un ensemble de traitements au niveau pragmatique du générateur, qui soit aussi générique que possible et en même temps ajustable pour un certain domaine d'application. Ainsi, on essait de définir de manière générique les attributs des usagers humains qui soient relevants à la génération pragmatique, mais, on le verra, aussi à la génération linguistique.

Pour l'approche pragmatique à la génération on doit forcément se limiter à un domaine d'application assez restreint. Ainsi, on se concerne avec la réservation et l'emprunt des livres à une bibliothèque.

Les caractéristiques du dialogue pour un tel domaine sont :

- **Le caractère spontané** - le dialogue est oral et l'intéraction entre l'usager et "la bibliothèque" est variée et influencée par les particularités d'expression de chaque utilisateur ;
- **Le caractère limité des buts** - l'ensemble des *buts* potentiels est très limité et réduit essentiellement à l'emprunt des livres ou bien à leur réservation, à l'apprentissage des informations à l'égard d'un certain livre, auteur ou domaine (du livre) ;

– **Le caractère temporel** - les buts précisés auparavant sont essentiellement sous-sommés à celui de l'établissement d'un *créneau* temporel où un certain livre (plus généralement : ouvrage) puisse être emprunté par un utilisateur de la bibliothèque ;
– **Le caractère négociatif** - les créneaux temporels en question sont sujet de négociation entre les usagers, par l'intermède du système gérant le fonctionnement de la bibliothèque : par exemple, lorsqu'un ouvrage est déja prêté à quelqu'un et il est en même temps souhaité pour réservation par deux autres clients, le système doit être capable de proposer des intervalles de temps d'accès à l'ouvrage de façon qu'il satisfasse tous les *trois* clients (celui ayant déjà emprunté l'ouvrage et les deux autres souhaitant le réserver) ;

En lumière de ces caractéristiques, la SDRT sera adaptée afin de répondre le mieux possible aux contraintes d'un tel dialogue, en gardant en même temps la généricité de l'approche et la simplicité de l'architecture.

La section suivante décrit les éléments particuliers choisis pour l'expression des formules logiques et l'ontologie discursive (générique par rapport à la tâche) ; la troisième section décrit l'approximation d'un fragment de la SDRT via la logique du premier ordre, intégrant un ensemble de sémantiques définies pour les relations rhétoriques et un algorithme pour la mise à jour des structures discursives SDRS ; la quatrième section présente un exemple détaillé de calcul d'une relation rhétorique mettant à l'épreuve l'ontologie discursive et les sémantiques des relations rhétoriques. Enfin, la cinquième section conclue le papier et relève des pistes pour des recherches dans l'avenir.

2 Préparatifs : formalisme logique et ontologies

2.1 Expression des formules logiques

Une formule logique, dans la manière la plus générale, a la forme α, exprimée dans la logique commune du premier ordre (CL – "Common Logic") et contenant cinq types d'information : (i) connectifs logiques : $\{\wedge, \vee, \neg, \Rightarrow\}$, (ii) quantificateurs : $\{\forall, \exists\}$, (iii) objets : $\langle \mathsf{name} \rangle$, (iv) prédicats : $\langle \mathsf{predicate\_name}(variable\_set) \rangle$.

Les entités du type (i) peuvent relier à leur tour d'autres formules logiques. Les entités du type (ii) peuvent préceder des entités du type (iii). Les entités du type (iii) peuvent être suivies par des entités du type (iv). Les entités du type (iv) sont précédées par des entités du type (iii).

Afin de pouvoir préciser en détail le contenu des formules logiques, on tient compte que le domaine d'application visé consiste, pour le moment, en dialogues auprès d'une bibliothèque, lors d'emprunt ou réservation de livres, et que les aspects temporaux sont essentiels pour le déroulement avec succès de ces dialogues. Ainsi, on introduit et emploie une taxinomie de *temps* possibles, afin d'augmenter l'expressivité de la logique commune du premier ordre. Ces marqueurs de temps possibles apparaîssent en tant qu'incices inférieurs droits et sont : (i) $t_{\#}$ — conditionnel présent, (ii) t_{+} — futur et "nouveau" / "jeune", (iii) t — présent, (iv) t_{-} — passé composé, imparfait et "vieux" / "ancien", (v) $t_{=}$ — plus que parfait, (vi) $t_{\pm}$ — conditionnel passé, (vii) $t_{\mp}$ — futur antérieur, (viii) $t_{\exists}$ — un moment unique qui n'est pas précisément situé sur l'axe du temps, (ix) $t_{\forall}$ — tout moment ; éternel, permanent.

2.2 Ontologie discursive

L'objectif de cette section est représenté par la conception de l'ontologie discursive pour la génération pragmatique. Les éléments de cette ontologie sont représentées par une base de connaissances pour les *sémantiques* des relations rhétoriques, dans le cadre de la SDRT.

La base de connaissances est employée par le générateur pour inférer les relations rhétoriques entre les énoncés[1].

Du point de vue **physique** la base de connaissances est structurée sous la forme de :
– une base de données en format textuel, XML ;
– un module logiciel pour la gestion des connaissances sous-tendues par la base de données, qui s'appuie sur la logique du premier ordre pour raisonner et a le rôle de gérer et interpréter les entités dans la base de données, afin que ces données deviennent *connaissances*.

Dans cette section on se propose de présenterles spécificités de la base de connaissances du générateur, en fonction du contexte discursif.

Une des limitations de la logique du premier ordre est la difficulté d'exprimer les **exceptions** aux règles / axiomes connues a priori. Quand même, une idée-clé pour la conception des bases de connaissances est l'organisation des objets en **catégories**, représentables dans la logique du premier ordre sous la forme de *prédicats* et *objets* (constantes), via la **réification** des prédicats. En plus, ces catégories peuvent organiser les bases de connaissances via le mécanisme de l'**héritage** (qui, d'ailleurs, peut être *multiple*).

Ainsi, pour toute base de connaissances, aussi générale qu'elle soit, on a un ensemble minimal de prédicats nécessaires : (i) "$\in$" — "MemberOf", (ii) "$\subset$" — "SubclassOf", (iii) "$\ni$" — "ClassOf", (iv) "$\supset$" — "SuperclassOf", (v) "$\cap = \emptyset$"

[1] Les énoncés sont exprimés dans la logique commune du premier ordre.

— “Disjoint”, (vi) “$\cup = \mathsf{All}$” — “ExhaustiveDecomposition”, (vii) “$(\cap = \emptyset) \wedge (\cup = \mathsf{All})$” — “Partition”, (viii) “$\bar{\in}$” — “SmallestMemberOf”, (ix) “$\bar{\subset}$” — “SmallestSubclassOf”[2].

Pour exprimer les **mesures**, on définit les prédicats suivants : “$<$” (“smaller”), “$>$” (“greater”), “$=$” (“equals”).

Pour la gestion des événements temporels on a un ensemble de prédicats spécifiques, inspirés des consignes de [21]. Ainsi, les prédicats, accompagnés par leurs sémantiques, sont montrés ci-dessous :

1. $\mathsf{Initiates}(\alpha, \beta, t) ::= \alpha \wedge (\mathsf{greater}(t_\alpha, t) \vee \mathsf{equals}(t_\alpha, t)) \Rightarrow \beta$; ce prédicat indique que le début de l'événement décrit par α au moment t implique la validité de la proposition β ;
2. $\mathsf{Terminates}(\alpha, \beta, t) ::= \alpha \wedge (\mathsf{smaller}(t_\alpha, t)) \Rightarrow \neg\beta$; ce prédicat indique que la fin de l'événement décrit par α au moment t implique la non-validité de la proposition β ;
3. $\mathsf{Clipped}(\alpha, t, t\prime) ::= \exists\beta : \mathsf{Terminates}(\beta, \alpha, t_\beta \wedge \mathsf{smaller}(t, t_\beta) \wedge \mathsf{smaller}(t_\beta, t\prime)$; ce prédicat indique que l'événement décrit par α est déterminé par un autre événement à un moment $\tau \in (t, t\prime)$;

Du point de vue “données”, chaque entité de l'ontologie est soit un **tableau** (si elle n'est pas *terminale*, dans le sens classique du terme), soit un **type** (une *collonne*) (si elle est terminale, partie d'un tableau, ou les deux). Ainsi, les connaissances sont structurées sous la forme d'une base de données relationnelle hiérarchique, où chaque **collonne** de tableau (donc, chaque *type*) devient **tableau**, dont les *types* (collonnes) sont les **instances** non-nulles du tableau supérieur du point de vue hiérarchique (donc, du tableau dont le tableau courant est un type / une collonne).

La base de données est parcourue jusqu'à ce que, pour un certain identifiant ID, on arrive à une valeur (instance) **terminale**, c'est à dire, qui n'est ni tableau, ni type / collonne dans un tableau.

Afin d'acquérir, au niveau logique, une représentation qui rend visible la qualité des données de représenter des *connaissances*, on **plonge** la hiérarchie établie au niveau “données” (XML) dans le formalisme de la logique du premier ordre.

Premièrement, on spécifie les relations d'inclusion entre les tableaux, types et instances dans l'ontologie : si on a que $Instance_k^{(ij)}$ est du type $Type_j^{(i)}$, alors on écrit $\mathsf{MemberOf}(Instance_k^{(ij)}, Type_j^{(i)})$, dans le cas où l'instance est *terminale*, et $\mathsf{SubclassOf}(Instance_k^{(ij)}, Type_j^{(i)})$ dans le cas où l'instance est non-terminale.

Donc, on arrive à un ensemble d'axiomes de **structuration** de l'ontologie, paramétrables selon les indices des types et des instances :

$\forall i, j, k : \forall l : \Big(\mathsf{MemberOf}(Instance_l^{(ijk)}, Type_k^{(ij)}) \wedge \mathsf{equals}(Instance_l^{(ijk)}, Type_l^{(ijk)})\Big) \vee$
$\Big(\mathsf{SubclassOf}(Instance_l^{(ijk)}, Type_k^{(ij)}) \wedge \mathsf{equals}(Instance_l^{(ijk)}, Type_l^{(ijk)})\Big)$.

En plus, on définit, de la même manière que ci-dessus, deux ensembles d'axiomes (ensembles, dans le même sens qu'auparavant) précisant le statut des instances par rapport aux types :

– si une instance est non-terminale, il existe des instances qui soient comprises dans la première :
$\exists i, j, k, l : \mathsf{equals}(Instance_l^{(ijk)}, Type_l^{(ijk)}) \Rightarrow \exists m : (\mathsf{MemberOf}(Instance_m^{(ijkl)}, Type_l^{(ijk)}) \wedge$
$\neg\mathsf{equals}(Instance_m^{(ijkl)}, Type_m^{(ijkl)}) \vee (\mathsf{SubclassOf}(Instance_m^{(ijkl)}, Type_l^{(ijk)}) \wedge$
$\mathsf{equals}(Instance_m^{(ijkl)}, Type_m^{(ijkl)})$;
– une instance est en même temps de deux (plusieurs) types (héritage multiple) si et seulement si elle inclue au moins une instance dont l'identifiant désigne deux instances, dans les deux types en question :
$\forall i, j, k : \Big(\mathsf{MemberOf}(Instance_k^{(ij)}, Type^{(ij)}) \wedge \mathsf{MemberOf}(Instance_k^{(ij)}, Type^{(ij)\prime})\Big) \vee$
$\Big(\mathsf{SubclassOf}(Instance_k^{(ij)}, Type^{(ij)}) \wedge \mathsf{SubclassOf}(Instance_k^{(ij)}, Type^{(ij)\prime})\Big) \Leftrightarrow$
$\exists m, m' : \Big(\mathsf{MemberOf}(Instance_m^{(ij)}, Type^{(ij)}) \wedge \mathsf{MemberOf}(Instance_{m'}^{(ij)}, Type^{(ij\prime)\prime})\Big) \vee$
$\Big(\mathsf{SubclassOf}(Instance_m^{(ij)}, Type^{(ij)}) \wedge \mathsf{SubclassOf}(Instance_{m'}^{(ij)}, Type^{(ij\prime)\prime})\Big) \wedge$
$\mathsf{equals}(ID(Instance_k^{(ij)}), ID(Instance_m^{(ij)})) \wedge \mathsf{equals}(ID(Instance_k^{(ij)}), Instance_{m'}^{(ij)})$.

La base des connaissances discursives est structurée dans notre architecture conformément aux consignes présentées auparavant, sauf un ensemble de particularités précisées ci-dessous. Ainsi, pour la structuration de la base de connaissances rhétoriques, l'ontologie comprend (hormis les entités précisées dans la sous-section antérieure) les entités suivantes :

– objets (constantes) : emitter, receiver, topic, enounce, question, Plan ;
– fonctions : $SARG()$, $Plan()$;
– prédicats : answer(), bad_time(), good_time().

Par conséquent, les connaissances sont structurées, au niveau logique, dans l'ontologie présentée dans la figure 1.

Sur la figure 1, les terminaux désignés par “$\{.....\}$” représentent des ensembles connus a priori, en vertu de la discussion antérieure, les terminaux désignés par “$\{\langle ...\rangle\}$” représentent des ensembles d'entités linguistiques (suites de mots, noms propres,

[2] Le dernier prédicat est défini, en termes des notations de la logique du premier ordre, via l'axiome $\forall y \forall x : \mathsf{MemberOf}(x, s) \Rightarrow \mathsf{SubclassOf}(x, y) \Rightarrow \mathsf{SubclassOf}(\mathsf{SmallestSubclassOf}(s, y), y)$.

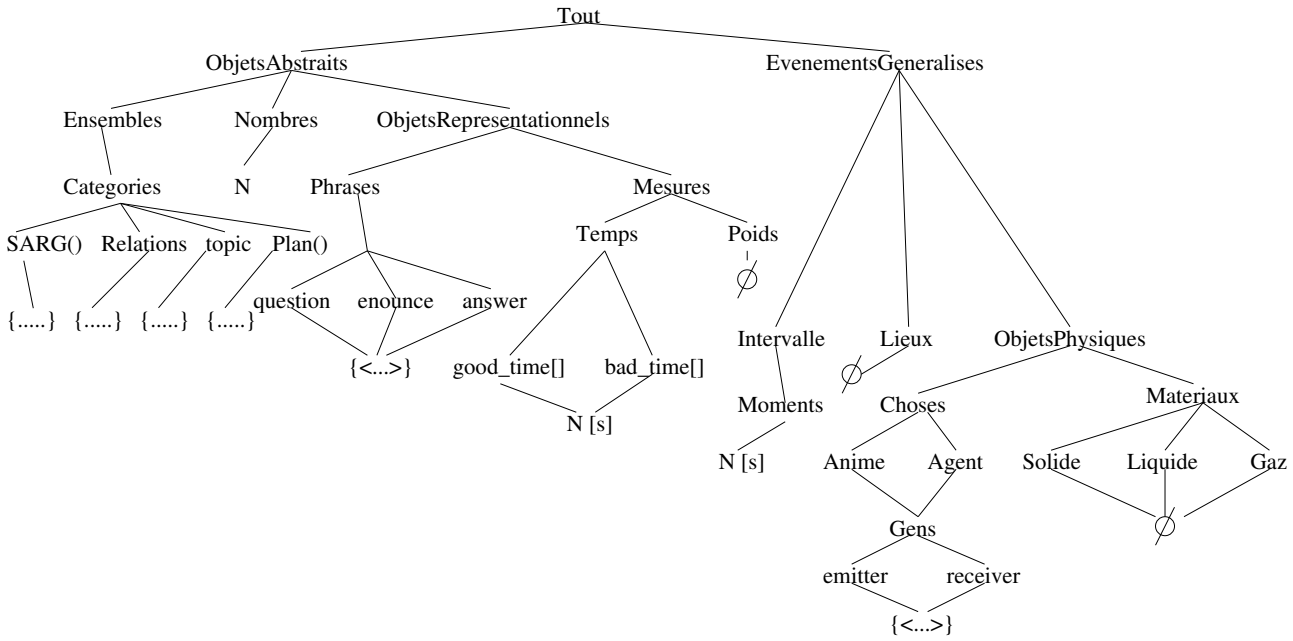

FIG. 1. Structuration des connaissances discursives, au niveau *logique*

...), la notation "()" attachée aux non-terminaux dénote le caractère de fonction, et la notation [] attachée aux non-terminaux dénote le caractère de prédicat ; enfin, la notation "N" désigne l'ensemble des entiers positifs, et "$\emptyset$" désigne l'ensemble vide.

Au niveau "données" les connaissances sont structurées toujours sous forme de tableaux, types et instances, selon les consignes précisées dans les sections antérieures.

Dans cette sous-section on va expliciter les prédicats et les fonctions employés lors de la définition des sémantiques formelles des 17 relations rhétoriques employées.

Le principe fondamental approché est de n'utiliser que la logique des prédicats du premier ordre, augmentée en expressivité via une ontologie discursive (présentée dans un chapitre ultérieur de ce document). On sait que la SDRS originale utilise la sémantique dynamique pour rendre compte des enchaînements rhétoriques dans un texte ; il est cependant difficile d'implémenter de façon directe ces logiques dynamiques, incluant des opérateurs modaux. C'est pour cela que de gros efforts ont été faits pour simplifier les sémantiques des relations rhétoriques. De ceux-ci, un des plus notables est l'essai de reformuler la SDRT dans la logique dynamique des prédicats (DPL – "Dynamic Predicate Logic") [23]. L'expression de la SDRT reste quand même encore carrément compliquée pour une réalisation informatique temps réel.

Dans le contexte fait voir ci-dessus et étant donné qu'en génération profonde les éléments de départ ne sont pas des morceaux textuels, mais des expressions logiques, il nous semble intéressant de simplifier encore davantage les sémantiques des relations rhétoriques dans le cadre de la SDRT, en s'appuyant seulement sur la logique des prédicats du premier ordre, pour laquelle il existe pas mal d'outils pour son traitement informatique (langages de programmation - PROLOG, démonstrateurs de théorèmes, etc.). Il y a quand même des problèmes avec cette approche, car dans la logique du premier ordre il n'est pas facile à quantifier les prédicats et les fonctions sur les objets ; en plus, le caractère non-monotone et défaisable des enchaînements rhétoriques n'est pas modélisable dans le cadre de la logique des prédicats seule. C'est pour cela qu'un ensemble de trucs & astuces est nécessaire ; pour nous, l'astuce est de s'appuyer sur une ontologie discursive à nombre fini et relativement petit d'instances pour les objets là-dedans. Les détails de conception et réalisation de cette ontologie sont présentés dans un chapitre ultérieur de ce document, mais essentiellement il s'agit d'une base de connaissances hiérarchique à double structure : premièrement, une base de données relationnelle spécifie les entités dans la base de connaissances et, deuxièmement, un ensemble d'axiomes exprimés toujours dans la logique du premier ordre spécifie les relations (de classification) entre ces entités.

Dans cette ontologie il y a des éléments, disons, terminaux (dans le sens où ces éléments n'ont plus de sous-types), pour lesquels il faut préciser des axiomes caractérisant les propriétés. C'est précisément ce qu'on va faire dans cette sous-section, notant par "Ω" l'ensemble des valeurs pour les objets dans l'ontologie discursive, par "$K(\alpha)$" la formule logique correspondant à l'énoncé α et par "t_α", le temps de production de α.

Il y a un ensemble de prédicats et fonctions primitifs dont la sémantique sera précisée :

1. $\mathsf{equals}(\alpha, \mathsf{question}) ::= \exists v : \mathsf{MemberOf}(v, K(\alpha)) \wedge \neg\exists\omega : \mathsf{MemberOf}(\omega, \Omega) \wedge \mathsf{equals}(v, \omega)$;
2. $\mathsf{equals}(\alpha, \mathsf{enounce}) ::= \neg\mathsf{equals}(\alpha, \mathsf{question})$;
3. $\mathsf{equals}(\beta, \mathsf{confirmation}(\alpha)) ::= \mathsf{equals}(K(\beta), K(\alpha)) \wedge \neg\mathsf{equals}(\mathsf{emitter}(\beta), \mathsf{emitter}(\alpha))$;

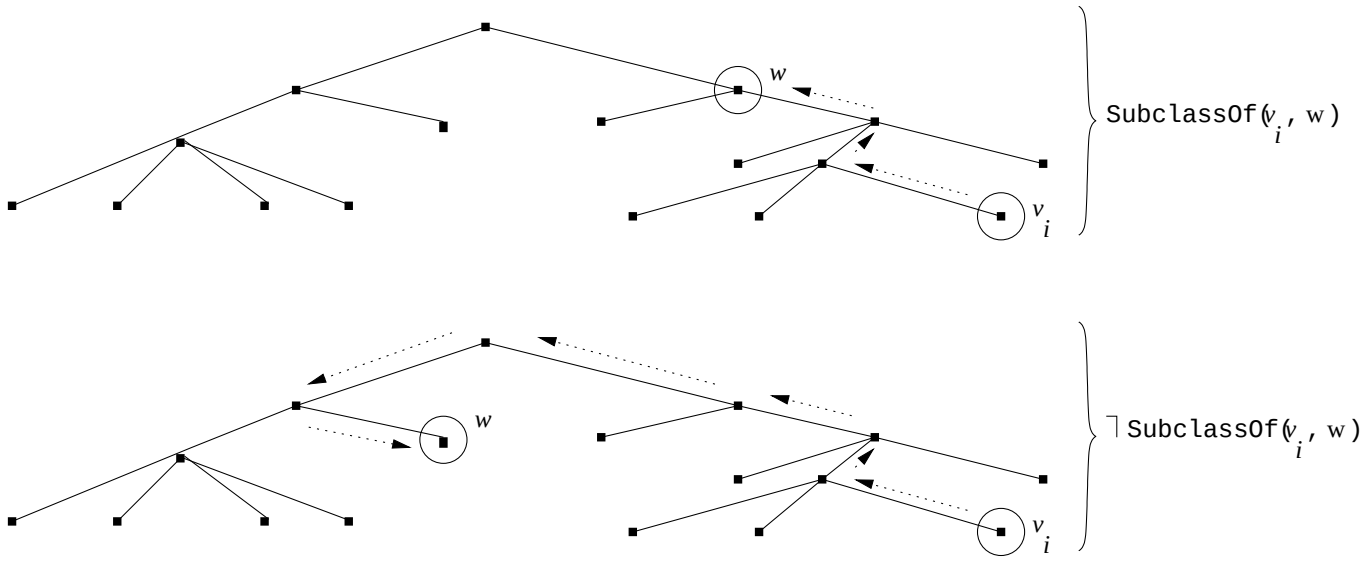

FIG. 2. Relations "ontologiques" entre les variables logiques

4. equals(β, answer(α)) ::= greater(t_β, t_α) $\wedge$ equals(topic(α), topic(β)) $\wedge \forall v$: MemberOf($v, K(\beta)$) $\Rightarrow$ $\exists\omega$: MemberOf(ω, Ω) $\wedge$ equals(v, ω) ;
5. 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($v_k, K(\alpha)$)) ;
6. emitter(α) ::= $(U \wedge \neg$Disjoint$(U, K(\alpha))) \vee (M \wedge \neg$Disjoint$(M, K(\alpha)))$;
7. entails(α, β) ::= $(K(\alpha) \Rightarrow K(\beta)) \wedge$ equals($t_\alpha, t_\forall$) ;
8. $SARG(\alpha)$ ::= $\tilde{K}(\alpha)$: SubclassOf($\tilde{K}(\alpha), K(\alpha)$) $\vee$ equals($\tilde{K}(\alpha), K(\alpha)$) $\wedge \exists\gamma$: MemberOf(γ, Γ)$\wedge$ $\neg$Disjoint($\gamma, \tilde{K}(\alpha)$;
9. good_time(Δt_β) ::= $\exists\alpha : \neg$Disjoint(topic(α), topic(β)) $\wedge$ smaller(t_α, t_β) $\wedge$ (SubclassOf($\Delta t_\beta, \Delta t_\alpha$)$\vee$ equals($\Delta t_\beta, \Delta t_\alpha$)) ;
10. bad_time(Δt_β) ::= $\neg$good_time(Δt_β) ;
11. Δt_β ::= $\forall t_1, t_2$: smaller(t_1, t_2) $\wedge$ MemberOf($t_2 \setminus t_1$, topic(β)) $\wedge$ MemberOf($t_1, K(\beta)$) $\wedge$ MemberOf($t_2, K(\beta)$).

Dans ces expressions formelles des prédicats et fonctions employés dans la définition des sémantiques discursives, Γ est l'ensemble des actes de langage. On observe que, contrairement à l'acception classique dans la SDRT, la notion de topique d'un énoncé est définie ici en termes d'ensemble des objets de l'ontologie du domaine (cf. chapitre 11) référés de manière déterminée[3] dans l'énoncé.

En plus, considérant que dans les expressions ci-dessus "v_i" et "w" sont des variables logiques, les relations "equals()" et "SubclassOf()" entre elles signifient, respectivement que les deux variables désignent le même objet dans l'ontologie du domaine ET qu'il existe, entre v_i et w un chemin à un seul sens et sans franchir deux fois le même point dans l'arbre de l'ontologie du domaine (un tel chemin s'appelle chemin hamiltonien). Pour mieux comprendre la relation "SubclassOf" entre les variables logiques désignant des objets dans l'ontologie, on va l'illustrer en principe dans la figure 2, où les flèches pointillées indiquent le chemin de coût minimal entre v_i et w.

Donc, les relations topiques entre les énoncés se déduisent à partir de l'ontologie du domaine, construite selon les consignes sur les bases de connaissances (l'ontologie).

3 Calcul de la structure discursive dialogique

3.1 La SDRT en dialogue homme-machine

La SDRT utilise la notion de *relation rhétorique*. Cette notion s'apparente à celle d'acte de langage mais n'en est pas équivalente. Elle se fonde plutôt sur la notion de paire adjacente issue des théories de la conversation dans laquelle tout acte

[3] Ici, un objet est défini de manière déterminée ssi la variable logique désignant l'objet de l'ontologie a une valeur permise attribuée dans la formule logique correspondante à l'énoncé.

de langage tente de “fermer” une paire ouverte. Cette vision du dialogue a cependant tendance à enfermer le dialogue dans une sorte de système de résolution des attentes. Cela renvoie à un modèle cognitif de “l’autre” puisqu’il s’agit maintenant, pour l’interlocuteur, d’interpréter ces attentes. De toutes façons, dans la perspective d’un modèle projectif, chaque acte est projeté vers le futur et prend sa signification dans un interacte construit de manière émergeante avec les autres acteurs dans le dialogue. Il s’agit donc plutôt de “projeter” le dialogue en avant à chaque instant, chaque acteur prenant sa part dans l’action mais aussi en en déléguant une partie à autrui. La mesure des effets de ces actions devient alors primordiale pour la poursuite du dialogue et la coordination mutuelle.

Pour ces raisons on a gardé la notion d’cte de langage (pour l’aspect projectif des buts qu’il offre) dans la modélisation du dialogue, et celle de relation rhétorique issue de la SDRT (“Segmented Discourse Representation Theory”) qui est plus compatible avec la gestion efficace de l’historique du dialogue en permettant de répercuter les effets des actions sur les attentes de manière réflexive.

Schématiquement tout se passe dans le “mental” d’un locuteur comme (i) s’il jugeait tout d’abord l’état de la situation - état du jeu, but à atteindre, connaissances nécessaires - (ii) s’il choisissait une solution de jeu - stratégie, acte - en accord avec les effets attendus sur autrui et sur la situation, (iii) qu’il produisait un acte dont une partie serait une délégation sous forme d’un interacte et qu’enfin (iv) il en évaluerait les effets par rapport à ses attentes.

Dans la SDRT, on étiquette les représentations des énoncés et les relations rhétoriques entre eux ; les dernières ont, bien sûr, comme arguments, les étiquettes des énoncés.

Une observation qui pour nous est cruciale est que, par différence avec la logique de Montague, en SDRT les représentations des entités(énoncés et relations rhétoriques entre eux) sont *indispensables*. Ces représentations (étiquettes) permettent de manipuler les instances d’énoncés (en particulier, pour le contexte abordé ici, les questions et les réponses) en tant que référents discursifs des actes de langage.

Pour toute relation discursive, il existe une interprétation dans le cadre de la sémantique dynamique, profusément présentée en [1]. Toutefois, pour nous cette représentation ne semble pas trop relevante en génération, donc, pour l’architecture du générateur pragmatique en soi. Par contre, ces interprétations des relations discursives sont plus relevantes pour le côté algorithmique du problème, car elles peuvent nous fournir des *critères* pour décider quelle relation rhétorique placer où (entre quels énoncés).

Ce qui semble intéressant dans le contexte architectural et du format de représentation pour les connaissances manipulées est la séparation que fait la SDRS entre la description du contenu informationnel de chaque énoncé “derrière” les étiquettes et la description de l’emballage informationnel, c’est à dire, des relations rhétoriques reliant les énoncés. Plus précisément, la SDRT utilise deux logiques différentes pour rendre compte des deux choses :

- Une logique du *contenu informationnel*, qui décrit du point de vue logique la structure interne conceptuelle des énoncés contenus dans une SDRS ou à ajouter à une SDRS ; cette logique est du premier ordre et *décidable*. Pour nos propres besoins, tenant compte de l’architecture choisie, on emploie la logique commune du premier ordre pour décrire le contenu des énoncés, donc, en fait, la logique dont le contrôleur de dialogue manipule et *engendre* les énoncés.
- Une logique de l’*emballage informationnel*, qui décrit du point de vue logique les relations rhétoriques entre les énoncés. Strictement du point de vue du générateur pragmatique, c’est *seulement* avec cette logique qu’il travaille. De toute façon, afin d’adapter cette logique au contexte dialogique *particulier* considéré ici, on lui ajoute un mécanisme qui gère la marche de la logique dans le cadre du dialogue. Ce mécanisme sera décrit plus loin.

Ceci nous permet d’utiliser la logique commune du premier ordre pour décrire le contenu propositionnel des énoncés.

La SDRT exprime le contenu d’un discours en général (donc, y compris le contenu d’un dialogue) en termes de structures discursives SDRS. Une SDRS peut être regardée à son tour comme une structuration *récursive*, via des relations rhétoriques, d’autres structures SDRS étiquetées (voir l’exemple de la section antérieure dans ce sens).

La mise à jour d’une SDRS se fait en principe en mettant en fonction une logique *non-monotone*[4] à l’aide de laquelle on *calcule* la relation rhétorique (vue comme un type d’acte de langage, selon la discussion d’auparavant) entre deux propositions (dans le sens logique du terme). Une telle relation rhétorique relie l’information nouvelle contenue dans un énoncé aux énoncés antécédents, donc, à l’historique du discours (dans notre cas, du dialogue).

Ces relations rhétoriques déduites via la logique de l’emballage informationnel imposent des contraintes à la fois sur le contenu propositionnel des énoncés que sur les SARG qu’ils véhiculent.

Dans l’approche classique de la SDRT, employée surtout pour l’*interprétation pragmatique*, les SARG sont récupérés du contexte discursif. Par contre, dans notre approche, les SARG des énoncés à engendrer sont définis par les actes de langage *renfermés* dans les formules logiques provenues du contrôleur de dialogue, comme l’on a vu dans le chapitre antérieur.

Donc, pour la génération, le problème n’est pas d’indentifier les SARG, mais de les traiter d’*a priori*s et de les *placer* dans un contexte discursif existant, donné par l’historique du dialogue.

Conséquemment, pour les buts fixés ici, on a choisi un ensemble de relations rhétoriques que le générateur pragmatique doit employer afin qu’il planifie le contenu du “quoi dire”. Ces relations rhétoriques se regroupent en trois catégories :

[4] Le caractère non-monotone peut signifier en logique beaucoup de choses, mais pour nous le seul sens relevant est celui de la variation de la valeur de vérité d’une proposition par rapport au temps discursif, donc, dans notre cas, par rapport aux tours de parole dans le dialogue.

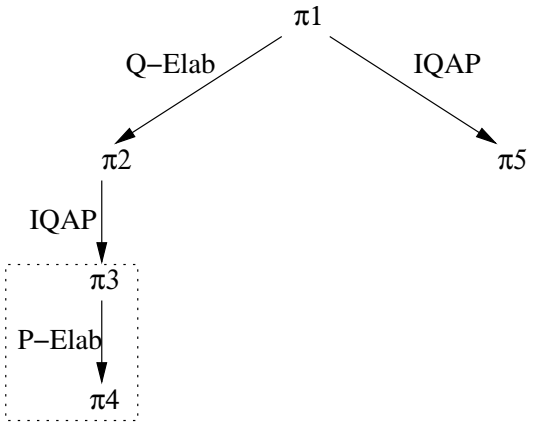

FIG. 3. La SDRS représentée sous forme de graphe

- Relations du **premier ordre** - il s'agit d'un ensemble de relations rhétoriques fortement liées aux aspects *temporels* du dialogue ; ces relations seront utilisées dans le générateur pragmatique d'une façon fortement approximative et particulière par rapport aux "prescriptions" de la SDRT ; ces relations sont : Q-Elab, IQAP, P-Corr et P-Elab, avec les sémantiques informelles précisées en [1] ;
- Relations du **second ordre** - il s'agit d'un ensemble de relations rhétoriques moins contraintes par le contexte temporel des dialogues approchés ; conséquemment, ces relations seront utilisées dans le générateur pragmatique dans une forme rapproché à celle établie par la SDRT [1] ; ces relations sont : Background$_q$, Elab$_q$, Narration$_q$, QAP, ACK et NEI ;
- Relations du **troisième ordre** - il s'agit d'un ensemble de relations rhétoriques non-dialogiques, qui puissent relier des énoncés à l'intérieur d'un seul tour de parole, provenus d'un seul locuteur dans le dialogue, soit U ou M ; ces relations sont : Alternation, Background, Consequence, Elaboration, Narration, Contrast et Parallel, de même, avec les sémantiques informelles telles que définies en [1].

Ces relations sont des types d'actes de langage, qui reflètent la dépendance entre le succès de (la performance de) l'acte courant et le contenu d'un ensemble d'actes antécédents.

On donne maintenant un court exemple illustratif pour les idées mises en lumière ci-desus, en considérant le dialogue ci-dessous, où les étiquettes $\pi_i,\ i = 1, ..., 5$ dénotent les répliques du dialogue, en fait leur représentation (dynamique - leur pertinence peut changer le long du déroulement du dialogue) :

M : π_1 : Vous voulez un livre sur le domaine 'X' de qui ?
U : π_2 : Mais, qu'est-ce que vous en avez disponible ?
M : π_3 : Nous avons l'auteur 'Y' et l'auteur 'Z'.
M : π_4 : On avait aussi 'T' et 'W' hier...
U : π_5 : Donc, je choisirais auteurs 'Z' et 'W' !

La structuration par des SDRS est *hiérarchique*, ce qui peut être mis en évidence par la représentation sous forme de graphe, comme dans la figure 3, où les flèches horizontales $\rightarrow$ indiquent des relations de coordonnation, tandis que les flèches (quasi-) verticales $\downarrow$ indiquent des relations de subordination.

On voit que dans cet exemple on n'a pas de relations de coordonnation, ce qui est assez typique pour le type de dialogue en question.

Dans le cadre de la SDRT on indentifie chaque relation rhétorique à un *type* d'acte de langage, en généralisant la notion de Searle d'acte de langage en soi. Conséquemment, on considère que le dialogue est *cohérent* du point de vue pragmatique si et seulement si pour tout énoncé il existe au moins une connexion entre celui-ci et un autre faisant partie de l'historique du dialogue, via une relation de discours.

Une telle relation de connexion entre deux énoncés sera appelée SARG ("Speech Act Related Goal"), donc, but en relation avec un acte de langage.

On ne fera ni une présentation introductive et qualitative des motivations de la SDRT pour la génération dans le contexte du dialogue hommme-machine, ni un compte rendu complet de la SDRT, car ceci est fait mieux qu'on ne puisse jamais le faire dans [1].

En revanche, on insistera surtout sur l'*adaptation* de la SDRS à nos buts : la génération dans le contexte du dialogue et sur des exemples d'utilisation.

Le matériau principal de travail dans notre contexte est constitué par les représentations discursives, qui font en fait l'objet du niveau pragmatique de traitement dans le générateur. Ainsi, suivant la SDRT, on étiquette les représentations des énoncés et les relations rhétoriques entre eux ; les dernières ont, bien sûr, comme arguments, les étiquettes des énoncés.

Une observation qui pour nous est cruciale est que, par différence avec la logique de Montague, en SDRT les représentations des entités(énoncés et relations rhétoriques entre eux) sont *indispensables*. Ces représentations (étiquettes) permettent de manipuler les instances d'énoncés (en particulier, pour le contexte abordé ici, les questions et les réponses) en tant que référents discursifs des actes de langage.

3.2 Approximations de la SDRT

Les approximations appliquées à la SDRT d'origine visent d'un côté la conservation aussi que possible de la SDRT classique (en raison de compatibilité à une théorie riche et ayant un fondement théorique solide) et, de l'autre côté, le maintien de la robustesse et de la précision, en termes d'*adéquation* de la théorie aux spécificités de notre problème : la génération dans le contexte d'un type particulier de dialogue homme-machine.

Comme l'on a déjà dit, on s'appuie sur le modèle dialogique de Jean Caelen, mais on ajoute deux suppositions supplémentaires, liées aux particularités du cadre de nos approches :

1. Les participants au dialogue (U et M) ne font pas consciemment de digressions par rapport à leur but, celui d'accèder à un livre 'X' au moment t ;
2. Conséquemment, tous les énoncés de U et M se rapportent au but précisé à 1., donc le SARG de tous les énoncés est d'offrir des informations sur :
   – la disponibilité d'un livre dans la bibliothèque, dans un créneau temporel précisé (souhaité par U) ;
   – le choix d'un certain livre, par rapport à certaines contraintes (domaine d'intérêt) et en foction de certains paramètres (auteur, anée de parution, etc.).

Donc, le bibliothécaire M et le lecteur U doivent se mettre d'accord sur les points suivants :
– un livre, en fonction du domaine, de l'auteur et de l'année de parution ;
– des intervalles de temps où le lecteur a accès au livre.
Donc, l'approximation de la SDRT se fait à deux niveaux :

Ici, on spécifie la connaissance des événements qui permettent l'accès à un livre à un moment donné, et de ceux qui ne le permettent pas. Donc, le système doit générer des réponses qui offrent des informations à l'égard de :

1. la description temporelle des créneaux disponibles, en relation avec le créneau Δt souhaité par l'utilisateur ;
2. la valence positive ou négative du créneau Δt, c'est à dire, si celui-ci est un "bon" créneau ou pas (autrement dit, si dans l'intervalle Δt l'utilisateur ayant fait la demande peut accéder au livre ou ne peut pas le faire) ;

Conséquemment, on *ignore*, de ce que le contrôleur de dialogue pourrait fournir au générateur pragmatique, des informations telles que :
– à qui se trouve le livre dans l'intervalle Δt souhaité par l'utilisateur ;
– la location précise du livre dans la bibliothèque (seulement si l'utilisateur ne souhaite en particulier se renseigner à cet égard) ;
– les raisons pour lesquelles le livre n'est pas disponible dans le créneau Δt, s'il ne l'est pas.

Donc, bref, le générateur pragmatique doit produire des réponses qui situent le créneau souhaité par U, Δt, en termes de temps propice pour l'emprunt ($\mathsf{good\_time}(\Delta t)$), ou pas propice pour l'emprunt ou la réservation ($\mathsf{bad\_time}(\Delta t)$). Il y a une autre approximation à ce même niveau : par exemple, si le temps Δt souhaité par l'usager est $\mathsf{good\_time}$, alors le générateur pragmatique peut transformer certaines formules logiques provenues du contrôleur de dialogue en questions à l'égard de Δt. Ainsi on a embarqué dans le générateur des règles qui font des "raccourcis" à la SDRT classique : donc, on évite des processus compliqués de raisonnement en créant des règles ad-hoc, appropriées pour notre application. Par exemple, cette règle-ci se formalise de la manière suivante :

$\mathsf{ClassOf}(\lambda^i_{\text{L.G.}}, \mathsf{good\_time}(\Delta t)) \Rightarrow (\exists j : \mathsf{smaller}(j,i) \wedge \exists R : \neg\mathsf{equals}(R\left(\lambda^i_{\text{L.G.}}, \lambda^j_{\text{L.G.}}\right), \emptyset) \wedge \mathsf{equals}(\lambda^i_{\text{L.G.}}, \mathsf{question}))$,

où $\lambda^i_{\text{L.G.}}$ est une formule logique provenu du générateur logique (L.G. - "Logic Generator") lors du tour de parole d'indice i (donc, on mesure ici le parcours du dialogue en termes de tours de parole ; cette approche est très utile pour l'implémentation).

Evidemment, en perspective la place que ces règles occupent dans le générateur devrait être cédée à l'engine de raisonnement de la SDRT, en raison de la généricité du système.

De toutes façons, pour une implémentation pratique faisable, un tel embarquement de règles ad-hoc est un choix de compromis entre la complexité du système et son adéquation au contexte concret de travail.

Du point de vue *discursif*, on suppose, selon le modèle de Jean Caelen, le caractère sincère des partenaires du dialogue, donc le SARG de transfert de croyance a toujours du succès. D'ici alors il résulte que s'il existe une question ayant la propriété que

la relation Q-Elab existe entre celle-ci et un énoncé de l'historique du dialogue, alors le générateur pragmatique ne crée jamais des énoncés visant le transfert de croyances entre U et M. Donc, on suppose que chaque agent est compétent par rapport à ses souhaits temporels et désirs de lecture, s'il s'agit de U, ou par rapport à la situation des livres de la bibliothèque (celui-ci est le travail du modèle de tâche), s'il s'agit de M.

Les règles pour le choix des relations discursives qui situent le "propos" du contrôleur de dialogue par rapport au contexte du discours sont monotones dans notre approche, tandis que dans la SDRT classique elles sont non-monotones. Ce choix est motivé par la possibilité d'éviter ainsi des vérifications de consistance coûteuses en termes de temps de calcul. Ces vérifications auraient été nécessaires lorsqu'on aurait utilisé des règles non-monotones, car la valeur de vérité des propositions inférées via ces règles aurait changé à chaque mise à jour du contexte discursif. Ici le caractère non-monotone est considéré au niveau de la convergence du dialogue, à travers des buts qui se rapprochent lors du déroulement du dialogue.

3.3 Sémantiques des relations rhétoriques

L'établissement *a priori* du but principal des interventions des locuteurs dans le dialogue, en relation avec les aspects temporels du dialogue, nous permet de préciser la sémantique des relations du premier ordre :

1. Q-Elab(α, β) : :=
   $\mathsf{equals}\ (\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{question}) \wedge \neg\mathsf{Disjoint}(\Delta t_\beta, SARG(\alpha)) \wedge (\forall\ \mathsf{answer}\ (\beta)) \neg \Rightarrow \neg SARG(\alpha)$;
2. IQAP(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{question}) \wedge \mathsf{equals}(\beta, \mathsf{question}) \wedge (K_\beta \Rightarrow \neg\mathsf{Disjoint}(\Delta t_\beta, SARG(\alpha)))$;
3. P-Corr(α, β) : :=
   $(K_\beta \Rightarrow \neg SARG(\alpha)) \Rightarrow (\mathsf{bad\_time}\ (\Delta t_\beta)) \wedge \mathsf{SuperclassOf}(\Delta t_\beta, SARG(\alpha))$;
4. P-Elab(α, β) : :=
   $\mathsf{good\_time}\ (\Delta t_\beta) \wedge \neg\mathsf{Disjoint}(\Delta t_\beta, SARG(\alpha)) \vee \mathsf{bad\_time}\ (\Delta t_\beta) \wedge \neg\mathsf{equals}(SARG(\alpha) \setminus \Delta t_\beta, \emptyset)$
   $\Rightarrow (\mathsf{ClassOf}(K_\beta, \mathsf{Plan})\ \wedge (\ \mathsf{Plan} \Rightarrow SARG(\alpha)))$;

De même, les sémantiques des relations de second ordre sont (ici, $K(\alpha)$ signifie le contenu sémantique, informationnel, de l'acte (énoncé) α) :

1. $\text{Background}_q(\alpha, \beta)$: :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge (\mathsf{equals}(\beta, \mathsf{question}) \wedge \forall\mathsf{answer}(\beta) \Rightarrow \text{Background}(\alpha, \mathsf{answer}(\beta)))$;
2. $\text{Elaboration}_q(\alpha, \beta)$: :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge (\mathsf{equals}(\beta, \mathsf{question}) \wedge \forall\mathsf{answer}(\beta) \Rightarrow \text{Elaboration}(\alpha, \mathsf{answer}(\beta)))$;
3. $\text{Narration}_q(\alpha, \beta)$: :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge (\mathsf{equals}(\beta, \mathsf{question}) \wedge \forall\mathsf{answer}(\beta) \Rightarrow \text{Narration}(\alpha, \mathsf{answer}(\beta)))$;
4. QAP(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{question}) \wedge \mathsf{equals}(\beta, \mathsf{answer}(\alpha)) \wedge \neg\mathsf{Disjoint}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta))$;
5. ACK(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \neg\mathsf{Disjoint}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta)) \wedge \mathsf{equals}(\beta, \mathsf{confirmation}(SARG(\alpha)))$;
6. NEI(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{question}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge (K(\beta) \Rightarrow \neg\mathsf{ClassOf}(\mathsf{emiter}(\beta), \mathsf{answer}(\alpha)))$.

Enfin, les sémantiques formelles pour les relations rhétoriques du troisième ordre sont :

1. Alternation(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \neg\mathsf{equals}(K(\alpha) \vee K(\beta), \emptyset)$;
2. Background(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \mathsf{entails}(\alpha, \beta)$;
3. Consequence(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge (K(\alpha) \Rightarrow K(\beta))$;
4. Elaboration(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \mathsf{SubclassOf}(\mathsf{topic}(\beta), \mathsf{topic}(\alpha)) \vee \mathsf{equals}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta))$;
5. Narration(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \neg\mathsf{Disjoint}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta)) \wedge \mathsf{smaller}(t_\alpha, t_\beta)$;
6. Contrast(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \neg\mathsf{Disjoint}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta)) \wedge \mathsf{entails}(\alpha, \neg\beta)$;
7. Parallel(α, β) : :=
   $\mathsf{equals}(\alpha, \mathsf{enounce}) \wedge \mathsf{equals}(\beta, \mathsf{enounce}) \wedge \neg\mathsf{Disjoint}(\mathsf{topic}(\alpha), \mathsf{topic}(\beta)) \wedge \mathsf{equals}(t_\alpha, t_\beta) \wedge \neg\mathsf{Disjoint}(K(\alpha), K(\beta))$.

3.4 Mise à jour de la SDRS dialogique

Le problème qu'on se pose ici est la mise à jour de la SDRS dialogique, au niveau du générateur pragmatique.

Plus précisément, étant donnée une formule logique K_α, provenue du contrôleur de dialogue, pour l'énoncé α à générer par la machine, on souhaite trouver l'ensemble $\Re$ de relations discursives et l'ensemble $\aleph$ d'énoncés ayant la propriété que : $\exists \Re(\alpha, \aleph)$.

Pour cela, notant par $\bar{\aleph}$ l'ensemble de *tous* les énoncés antérieurs à α dans le dialogue courant, on cherche, pour chaque $(\beta_i)_{i=\overline{1,|\bar{\aleph}|}}$ les relations rhétoriques ρ ayant la propriété que $\Sigma_\rho\left(K(\alpha), K(\beta_i)\right) \neq \mathsf{FALSE}$, où "$\sigma$" dénote l'opération de permutation et la dernière expression signifie que la formule logique obtenue en remplaçant ρ par sa sémantique formelle (Σ_ρ), avec les arguments α et β_i, dans les deux ordres possibles, et α et β_i par $K(\alpha)$ et $K(\beta_i)$, est *non-fausse*[5].

Donc, l'algorithme général d'opération pour la mise à jour de la SDRS dialogique est présenté ci-dessous, en employant les notations introduites auparavant :

1. pour l'énoncé libellé α fourni par le contrôleur de dialogue :
   (a) faire la substitution $\alpha \mapsto K(\alpha)$, en interrogeant le contrôleur de dialogue[6] ;
   (b) pour chaque énoncé $\beta_i \in \bar{\aleph}, i = \overline{1, |\bar{\aleph}|}$:
      i. pour chaque relation rhétorique ρ_j connue par le générateur ($j = \overline{1, 17}$) :
         A. faire la substitution $\rho_j \mapsto \Sigma_j$;
         B. calculer la valeur de vérité de :
            $\Sigma_j\left(\sigma\left(\alpha, \beta_i\right)\right), j = \overline{1, 17}, i = \overline{1, |\bar{\aleph}|}$ et note-la par γ ;
         C. si $\gamma = \mathsf{FALSE}$, alors passer à 1.(b).i. en incrémentant j ;
            autrement, sauvegarder ρ_j dans $\Re$ et β_i dans $\aleph$ et passer à 1.(b).i. en incrémentant j ;
         D. faire les attributions : $\aleph_\alpha \leftarrow \aleph$ et $\Re_\alpha \leftarrow \Re$;
2. vérifier la valeur de vérité pour la proposition $(\aleph_\alpha = \emptyset) \wedge (\Re_\alpha = \emptyset)$ et noter-la par υ :
   (a) si $\upsilon = \mathsf{FALSE}$, alors :
      i. ajouter α aux énoncés de la SDRS courante ;
      ii. ajouter $\Re_\alpha$ aux relations rhétoriques dans la SDRS courante ;
   (b) autrement (si $\upsilon = \mathsf{TRUE}$), alors construire une nouvelle SDRS ayant α en tant que seul énoncé et l'ensemble de relations rhétoriques vide.

Cet algorithme construit une structure discursive *complète* dans le sens que toutes les relations rhétoriques entre paires d'énoncés sont trouvées, sans appui sur un principe de maximisation de la cohérence (appliqué dans le cadre de la SDRT classique pour garder l'ensemble des relations rhétoriques reliant un nombre maximal d'énoncés, sans y avoir des contradictions) ; les contradictions (revenant au fait qu'une relation rhétorique soit révisée par des relations rhétoriques conséquentes) sont évitées supposant que la logique employée est monotone. Cela conduit, en effet, à l'obtention de toutes les structures SDRS potentielles, sans en repérer la structure la plus cohérente (dans le sens de la SDRT classique).

4 Exemple de calcul d'une relation rhétorique

Dans cette sous-section on va illustrer les traitements accomplis lors de la réalisation de l'étape 1.(b).i.B. de l'algorithme général pour la mise à jour de la SDRS dialogique. En d'autres mots, on va montrer, sur un exemple concret, comment on calcule la valeur de vérité de la proposition logique $\neg\left(\Sigma_j\left(\sigma\left(\alpha, \delta_i\right)\right)\right), j = \overline{1, 17}, i = \overline{1, |\aleph|}$, où "$\aleph$" est l'ensemble des énoncés (en tant que *formules logiques*) disponibles pour les faire joindre une SDRS.

Plus précisément, on suppose qu'on a les énoncés :

$U : \alpha$: Je lirai ce livre lundi.
$M : \beta$: Est-ce bien pour vous à 14 h ?

En fait, l'énoncé "α" est provenu de l'utilisateur, donc disponible à la fois sous forme textuelle que sous forme logique (calculée par l'analyseur sémantique et l'interpréteur pragmatique), tandis que "β" n'est disponible que sous forme logique, calculée par le contrôleur de dialogue ; sa forme textuelle est à obtenir par le générateur. En particulier, entre ces deux énoncés, c'est bien la relation Q-Elab qui existe entre α et β. Néanmoins, le générateur ne peut le savoir *a priori*, donc il doit essayer chaque relation des 17 définies ; on suppose, pour la simplicité de l'exposé, que c'est justement la relation Q-Elab(α, β) qu'on vérifie. Pour cela, il faut parcourir les étapes suivantes :

[5] Si une proposition logique est "non-fausse", cela ne signifie pas forcément qu'elle soit vraie.

[6] Pour des détails à l'égard de la structure et la gestion, du point de vue du générateur, des formules logiques, voir le chapitre suivant, portant sur la génération linguistique.

1. Retrouver les sémantiques de α et β :
   - $\alpha \mapsto K(\alpha) \mapsto \exists X, Y$: object(X) $\wedge$ equals(X, book) $\wedge$ agent(Y) $\wedge$ equals(Y, U) $\wedge$ equals(t_α, t) $\wedge$ read(Y, X) $\wedge$ equals($\Delta t_\alpha, t_+$) ;
   - $\beta \mapsto K(\beta) \mapsto \exists X, Y$: object(X) $\wedge$ equals(X, book) $\wedge$ agent(Y) $\wedge$ equals(Y, U) $\wedge$ read(Y, X) $\wedge$ greater(t_β, t_α) $\wedge$ equals(Δt_β, '14h') $\wedge$ equals($\Delta t_\beta, t_+$) $\wedge$ equals(good_time(Δt_β), ?) ;

2. Retrouver la sémantique de la relation rhétorique vérifiée couramment :
   $j \mapsto \Sigma_j \mapsto \Sigma_{\text{Q-Elab}} \mapsto$ equals (α, enounce) $\wedge$ equals(β, question) $\wedge \neg$Disjoint($\Delta t_\beta, SARG(\alpha)$)$\wedge$
   ($\forall$ answer (β) $\neg \Rightarrow \neg SARG(\alpha)$) ;

3. Expliciter chaque prédicat et fonction dans la sémantique de la relation rhétorique vérifiée couramment :
   - equals(α, enounce) $\mapsto \neg$equals(α, question) $\mapsto \forall v$: MemberOf($v, K(\alpha)$) $\vee \exists \omega$: MemberOf(ω, Ω) $\vee$ equals(v, ω) ;
   - equals(β, question) $\mapsto \exists v'$: MemberOf($v', K(\beta)$) $\wedge \neg \exists \omega'$: MemberOf(ω', Ω) $\wedge$ equals(v', ω') ;
   - $SARG(\alpha) \mapsto \exists X, Y, \theta$: object(X) $\wedge$ equals(X, book) $\wedge$ agent(Y) $\wedge$ equals(Y, U) $\wedge$ good_time(θ) $\wedge$ equals($\theta, \Delta t_\alpha$) $\wedge$ greater(θ, 'lundi') ;
   - $\Delta t_\beta \mapsto$ '14h' $\wedge$ 'lundi';
   - $\forall$answer(β) $\mapsto \forall \delta$: equals(δ, answer(β)) $\mapsto \forall \delta$: greater(t_δ, t_β)$\wedge$equals(topic(δ, topic(β))$\wedge \forall v''$: MemberOf($v'', K(\delta)$) $\Rightarrow$ $\exists \omega''$: MemberOf(ω'', Ω) $\wedge$ equals(v'', ω'') ;
   - $\neg SARG(\alpha) \mapsto \forall X, Y, \theta$: object(X) $\vee$ equals(Xbook) $\vee$ agent(Y) $\vee$ equals(Y, U) $\vee$ bad_time(θ) $\vee$ equals($\theta, \Delta t_\alpha$) $\vee$ greater(θ, 'lundi') ;

4. Expliciter chaque prédicat et fonction qui apparait dans les prédicats et fonctions explicités précédemment :
   - topic(β) $\mapsto$ ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+) ;
   - good_time(θ) $\mapsto \exists \gamma : \neg$Disjoint(topic($\gamma$), topic($\pi$)) $\wedge$ smaller(t_α, t_π) $\wedge$ (SubclassOf($\theta, \Delta t_\alpha$)$\vee$ equals($\theta, \Delta t_\alpha$)) $\wedge \pi$: equals($\Delta t_\pi, \theta$) ;

5. Calculer la valeur de vérité de chaque proposition logique renfermée entre conjonctions ("$\wedge$"), dans la sémantique de la relation rhétorique vérifiée couramment :
   - equals(α, enounce) $\longrightarrow$ TRUE ; cela est calculé substituant la sémantique de α dans la sémantique de la proposition ci-contre et explorant, dans l'ontologie discursive ET l'ontologie du domaine (Ω), toutes les valeurs possibles pour les variables concernées ;
   - equals(β, question) $\longrightarrow$ TRUE ; cela est calculé de la même manière que ci-dessus ;
   - $\neg$Disjoint($\Delta t_\beta, SARG(\alpha)$) $\longrightarrow$ TRUE ; cela est résulté directement de la proposition :
     SubclassOf($\Delta t_\beta, \Delta t_\alpha$) $\wedge$ SubclassOf($\Delta t_\alpha, SARG(\alpha)$) ;
   - $\forall$ answer (β) $\neg \Rightarrow \neg SARG(\alpha) \mapsto \forall \delta$: equals(δ, answer(β)) $\mapsto \forall \delta$: greater(t_δ, t_β)$\wedge$
     equals(topic(δ), ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+))$\wedge$
     $\forall v''$: MemberOf($v'', K(\delta)$) $\Rightarrow \exists \omega''$: MemberOf(ω'', Ω)$\wedge$equals(v'', ω'')$\neg \Rightarrow \forall X, Y$: object(X)$\vee$equals(X, book)$\vee$
     agent(Y) $\vee$ equals(Y, U) $\vee$ bad_time(Δt_β) $\vee$ equals($\Delta t_\beta, \Delta t_\alpha$) $\vee$ greater(Δt_β, 'lundi') ;
     la dernière formule a été obtenue en substituant les fonctions et les prédicats par leurs définitions explicites, et la variable θ, par Δt_β ; ensuite, par la *skolémisation*[7] des variables dans cette dernière formule, on obtient :
     $\forall$ answer (β) $\neg \Rightarrow \neg SARG(\alpha) \mapsto \forall \delta$: equals(δ, answer(β)) $\mapsto \forall \delta$: greater(t_δ, t_β)$\wedge$
     equals(topic(δ), ExhaustiveDecomposition(book, read, good_time('14h'), good_time('lundi'), t_+))$\wedge$
     $\forall v''$: MemberOf($v'', K(\delta)$) $\Rightarrow$ MemberOf(ω_0, Ω) $\wedge$ equals(v'', ω_0)$\neg \Rightarrow \forall X, Y$: object(X) $\vee$ equals(X, book) $\vee$
     agent(Y) $\vee$ equals(Y, U) $\vee$ bad_time(Δt_β) $\vee$ equals($\Delta t_\beta, \Delta t_\alpha$) $\vee$ greater(Δt_β, 'lundi') ;
     ensuite, on élimine les quantificateurs universels, et on explore les voies possibles dans l'ontologie du domaine, obtenant non-contradiction, donc la valeur "TRUE" ;

Cet exemple montre qu'il est possible, en principe, de déterminer une relation rhétorique reliant un énoncé courant, à générer, disponible sous forme logique seulement, à un dialogue en déroulement, via l'utilisation de la logique du premier ordre, renforcée par une ontologie discursive indépendante de tâche. Il reste quand même à évaluer la complexité de ces calculs, étudiant des possibilités de réduire le temps de calcul en complétant le formalisme logique utilisé avec des opérateurs modaux, par exemple. Dans ce dernier sens, on peut voir que l'essai d'exprimer de manière rigoureuse la sémantique de la relation Narration dans le cadre de la logique dynamique des prédicats a besoin d'un opérateur modal (celui de "nécessité par défaut") [23] ce qui dépasse les possibilités expressives de la logique dynamique des prédicats. Pour cela, il peut être envisagé de compléter ces formalismes avec des opérateurs modaux.

[7] Le processus de *skolémisation* signifie l'élimination des quantificateurs existentiels ; ceci est fait identifiant des valeurs dans le domaine de définition des variables quantifiées avec "$\exists$" et remplaçant ces variables avec les valeurs identifiées. Ces valeurs identifiées s'appellent des constantes skolem et seront, dans notre exemple, indiciées par "0", "1", etc..

5 Conclusions et perspectives

Les traitements pragmatiques représentent un niveau-clé dans l'architecture proposée ici, car le but est de rendre un système de dialogue aussi pertinent et expressif que possible, du point de vue de la génération des réponses à l'usager.

C'est pour cette raison que le générateur pragmatique (qui réalise les traitements au niveau pragmatique de génération) s'appuie sur une théorie formelle puissante, au moins pour la compréhension et l'interpétation du langage, la SDRT, pour la gestion *globale* de la cohérence des réponses générées par rapport au contexte dialogique.

Cet outil formel a été quand même fortement adapté, de façon ad-hoc, aux buts et au contexte de nos recherches, donc, à la génération en dialogue, visant en même temps la cohérence et la pertinence des réponses et la généricité[8] des démarches effectuées. Ainsi, les outils formels ont été adaptés et augmentés par rapport aux points suivants :

1. la spécification de sémantiques adéquates pour les relations discursives, dans le cadre de la SDRT ;
2. des mécanismes adéquats pour la mise à jour du contexte discursif (dialogique).

Afin de rendre aussi "concret" que possible ce module de traitements pragmatiques en génération pour le dialogue on a proposé aussi un ensemble d'algorithmes pour la gestion (vérification et mise à jour) de la structure discursive du dialogue .

Un enjeu important qu'on s'est proposé a été de n'utiliser que la logique des prédicats du premier ordre, évitant ainsi les sémantiques dynamiques formalisant les relations rhétoriques et les sens des énoncés dans la SDRT classique. L'indécidabilité entraînée par le caractère non-monotone des enchaînements rhétoriques est évitée dans notre approche utilisant une ontologie hiérarchique guidant les raisonnements logiques. Cette ontologie est organisée à son tour sur deux niveaux – un niveau données et un niveau axiomatique. Le couplage carrément étroit entre les sémantiques des relations rhétoriques et l'ontologie rend le calcul des preuves logiques décidable et faisable du point de vue computationnel. La raison pour ce choix du formalisme logique réside dans sa simplicité et le support logiciel existant (outils de raisonnement automatique, environnements de programmation – PROLOG, etc...). Un protoype, comprenant l'ensemble des traitements décrits dans cet article, est en train d'être implémenté en GNU PROLOG.

Sachant qu'un dialogue peut arriver à comprendre un ensemble assez étendu de tours de parole et que l'ensemble de relations rhétoriques dialogiques comprend 10 éléments, calculer la valeur de vérité pour chaque relation rhétorique impliquant la dernière contribution dialogique et chacune des contributions antérieures peut être très coûteux en temps de calcul. Ceci dit, il est important de rendre aussi efficace que possible le générateur pragmatique, en opérant certaines optimisations dans les calculs. Les optimisations envisagées sont de nature algorithmique, de principe et touchent deux volets :

1. réduire le nombre de tours de parole candidats précédant le tour de parole courant, à relier via des relations rhétoriques au contexte du dialogue ;
2. pour une paire d'énoncés, réduire l'ensemble des relations rhétoriques à "essayer", à un sous-ensemble de relations rhétoriques envisageables / possibles entre une paire d'énoncés en tant qu'actes de langage.

Le premier volet peut être satisfait en limitant la taille de la pile des actes de langage (voir ci-dessus) à N tours de parole. Compte tenu des démarches en recherche des aspects cognitifs du discours, une valeur $N = 7$ modélise de manière satisfaisante des limites de la "mémoire discursive", c'est à dire un locuteur produit une contribution dialogique orale en relation avec un sous-ensemble des 7 derniers tours de parole [21]. Conséquemment, on peut considérer dans le calcul des relations rhétoriques seulement la contribution courante et les 7 contributions antérieures ; si aucune relation rhétorique ne peut être calculée entre ces énoncés, alors on accroît la valeur de N jusqu'à ce qu'au moins une relation rhétorique est trouvée entre la contribution courante et les répliques antérieures.

Pour satisfaire le deuxième volet, il faut d'abord annoter un corpus de dialogues en actes de langage et relations rhétoriques et, ensuite, voir si pour une paire d'actes de langage (soit en tant que tours de parole distincts en dialogue, soit en tant qu'énoncés dans le cadre d'un même tour de parole) on peut dégager un sous-ensemble de relations rhétoriques possibles. Il existe cependant une étude dans ce sens [26], mais là on trouve une taxinomie des relations rhétoriques légèrement différente de la nôtre. En plus, dans l'étude citée les correspondances (paire d'actes de langage) $\mapsto$ (relation rhétorique) sont illustrées via des exemples construits de manière ad-hoc et non pas tirés d'un corpus réel de dialogues. C'est pour cette raison qu'on va d'abord annoter un corpus de dialogues réels et ensuite établir les correspondances entre les paires d'actes et les relations rhétoriques. Ainsi on pourra optimiser les performances de l'opération *en ligne* du générateur pragmatique, lors de la construction dynamique d'une SDRS dialogique le long d'une interaction langagière entre l'utilisateur et la machine.

Références

1. ASHER, N., LASCARIDES, A., *Logics of Conversation*, Cambridge University Press, 2003.

[8] Ici par "généricité" on comprend la possibilité de paramétrer un mécanisme général de gestion via des politiques ("policies") adaptées au domaine d'utilisation et à l'application. Donc, on ne confond pas les notions de "généralité" - utilisation du même système, tiré de la boîte, pour toute application, et "généricité" - la possibilité de paramétrer un mécanisme général pour une application particulière.

2. BILANGE, E., *Dialogue personne–machine*, Ed. Hermès, Paris, France, 1992.
3. CAELEN, J., IMBERDIS, L., "Génération des actes illocutoires pour le dialogue", *Actes du colloque GAT'97*, Grenoble, France, 1997.
4. CAELEN, J., XUEREB, A., *Interaction et pragmatique - jeux de dialogue et de langage*, Editions Hermès Lavoisier, Paris, 2007.
5. CARBERRY, S., LAMBERT, L., "A Process Model for Recognizing Communicative Acts and Modeling Negotiation Subdialogues", *Computational Linguistics*, vol. 25, no. 1, 1999.
6. CHAMBREUIL, M. (Ed.), *Sémantiques*, Ed. Hermès, Paris, France, 1998.
7. CHU-CARROLL, J., CARBERRY, S., "Collaborative Response Generation in Planning Dialogues", *Computational Linguistics*, vol. 24, no. 3, 1998.
8. DANLOS, L., EL-GHALI, D., "A complete integrated NLG system using AI and NLU tools", *Proceedings of COLING'02*, Taiwan, 2002.
9. DANLOS, L., GAIFFE, B., ROUSSARIE, L., "Document Structuring à la SDRT", *Proceedings of the International Workshop on Generation, ACL*, Toulouse, 2003.
10. LASCARIDES, A., OBERLANDER, J., "Abducing Temporal Discourse", *Aspects of Automated Natural Language Generation*, Springer, 1992.
11. LUGER, G. F., *Artificial Intelligence : Structures and Strategies for Complex Problem Solving*, Addison-Wesley, 2005.
12. MAUDET, N., MULLER, Ph., PREVOT, L., "Tableaux conversationnels en SDRT", *Actes de TALN 2004*, Fès, Maroc, 2004.
13. MCTEAR, M. F., "Spoken Language Technology : Enabling the Conversational User Interface", *ACM Computing Surveys*, vol. 34, no. 1, 2002.
14. MINKER, W., BENNACEF, S., *Parole et dialogue homme–machine*, Editions CNRS – Eyrolles, 2001.
15. NGUYEN, H., *Dialogue homme–machine : Modélisation de multisession*, Thèse de doctorat, Université Joseph Fourier, Grenoble, France, 2005.
16. NILSSON, U., MALUSZYNSKI, J., *Logic, Programming and Prolog*, John Wiley & Sons, second edition, 1995.
17. PEREIRA, F., SHIEBER, S., *Prolog for Natural Language Analysis*, CSLI, 1987.
18. POPESCU, V., "Architecture pour la génération du langage naturel en dialogue homme-machine", *Ecole thématique du CNRS "Dialogue et Interaction"*, Autrans, France, 2006.
19. REITER, E., DALE, R., *Building Natural Language Generation Systems*, Cambridge University Press, 2000.
20. ROUSSARIE, L., *Un modèle théorique d'inférence de structures sémantiques et discursives dans le cadre de la génération automatique de textes*, Thèse de doctorat, Université Paris 7, Paris, France, 2000.
21. RUSSELL, S., NORVIG, P., *Artificial Intelligence : A Modern Approach*, Prentice Hall, 2003.
22. SCHLANGEN, D., LASCARIDES, A., COPESTAKE, A., "Resolving Underspecification using Discourse Information", *Proceedings of BI-DIALOG 2001*, Springer, 2001.
23. STAUDACHER, M., *SDRT Reformulated using DPL*, Term Paper, Bielefeld University, Germany, 2005.
24. VANDERVEKEN, D., *Meaning and Speech Acts*, Cambridge University Press, 1990–1991.
25. VILLASENOR-PINEDA, L., *Contribution à l'apprentissage dans le dialogue homme-machine*, Thèse de doctorat, Université Joseph Fourier, Grenoble, 1999.
26. XUEREB, A., CAELEN, J., "Actes de langage et relations rhétoriques en dialogue homme–machine", *Séminaire Logique et Dialogue*, France, 2004.