Extraits du cahier des charges pour dégager éventuellement principes et définitions

la conception globale du schéma est « évolutive », elle n'est en rien figée en début de la réalisation (construction du schéma) ; le schéma doit pouvoir - tout au long de sa construction, et après - être révisable ;

mais le retour sur la définition des objets (attributs, corpus, schémas, contraintes) -leurs modifications- pose alors la question de l'hérédité : la rupture d'une certaine procédure ordonnant la construction du schéma (définition des attributs – construction du corpus- structure du schéma - définition des contraintes) peut rompre les liens entre les objets (attributs, corpus, schémas, contraintes) assurant la cohérence du schéma

La représentation en intention du schéma à produire dans le formalisme des graphes conceptuels devrait nous offrir beaucoup de possibilités. Mais, comme il est rappelé plus haut, il existe des liens de dépendances entre schémas, corpus, attributs et contraintes. Mon idée est de passer par le truchement de règles pour aller du "quoi" (l'expression du schéma du texte à produire) au "comment" (la procédure et les requêtes appliquées sur les attributs natifs ou calculés dans les corpus). Dans ce modèle on doit pouvoir contrôler la faisabilité d'un schéma, et mesurer l'impact d'une modification.

Les corpus pourront être externes (c'est-à-dire importés dans le système à partir de dictionnaire ou d'une base de données textuelles : liste d'expressions toutes faites, de fragments de dialogues, syntagmes, unités narratives, épisodes, etc.) ou internes (c'est-à-dire créées par l'utilisateur lui-même). L'importation – hors création de schéma – d'un corpus externe comportera nécessairement une phase de formatage (sous le contrôle de l'utilisateur) permettant d'associer aux items du corpus la suite des valeurs des attributs qui leur sont attachés et qui seront utilisés au moment de la mise en action des contraintes, c'est-à-dire dans la phase de production de l'œuvre.

Chaque corpus fournit un ensemble d'attributs. Dans la configuration actuelle, ces attributs sont définis "en dur", c.a.d. directement attachés au bout de texte qui constitue l'entrée principale. Si on conserve cette limitation, l'utilisateur aura tendance à constituer un corpus adhoc réalisé à la seule fin de satisfaire sa contrainte. Et forcément des attributs ne sont utilisés que dans un seul corpus pour une seule contrainte.
Sans doute la notion de corpus doit garder son niveau d'abstraction pour l'utilisateur mais elle devra, au niveau de l'implémentation, se décliner sous plusieurs formes. On ne traitera pas un corpus de quelques dizaines d'éléments (voire d'un seul) comme peut l'exiger la production de la même façon qu'on traitera un grand corpus. Ainsi on peut envisager d'utiliser des attributs calculés à partir des autres attributs et du texte d'un élément de corpus. On dispose pour cela d'un langage scripté dont je parle plus bas.

3.2.2.3. Objectifs sur les attributs

3.2.2.3.1.-Extension des types d'attributs utilisables : verbal, textuel, numérique, logique, typographique, autre...
3.2.2.3.2.-Les contraintes dites simples (relation entre des attributs) pourront être de différents types :

- indépendantes du type des attributs : égalité, inégalité, appartenance (des valeurs des attributs à une liste définie)
- ensemblistes : attributs compatibles (ayant au moins un élément commun), attributs incompatibles (deux à deux disjoints)
- numériques : le premier vaut 1 de plus (ou de moins) que le suivant, le

premier est la somme des suivants.

3.2.2.3.3. -Mises en place de contraintes dites conditionnelles, de la forme :

SI <contrainte> ALORS <contrainte> SINON <contrainte conditionnelle>

Les attributs externes (ou préétablis) seront empruntés aux domaines de la grammaire ou à des généralisations naturelles : genre, nombre, personne, champ thématique, aspect, caractéristique rhétorique ou stylistique, etc. Leur valeur pourra être celle-là même, convenablement abrégée, que leur donne l'usage : masc., fem., sing., pers1, prés-ind., etc. Les attributs créés par l'utilisateur pourront être identifiées et recevoir des valeurs choisies de façon à en traduire efficacement la signification et l'emploi.

Les relations externes (préétablies) seront, classiquement :

identique à
différent de
compatible avec

Dans le cas d'attributs numériques on pourra évidemment utiliser les relations ensemblistes ou algébriques usuelles : < , >, +, -, x et définir des relations de compatibilité ou d'incompatibilité spécifiques.
Ici encore, les relations définies par l'utilisateur se réfèreront de façon logique aux particularités des attributs pour lesquels ils seront pertinents. Ils devront correspondre à des opérations disponibles (donc déjà programmées) dans le système.
Dans la programmation du logiciel, les contraintes ne devront pas être vérifiées après l'exploration combinatoire mais en cours de cette exploration.

En dehors des opérations « naturelles » entre attributs, qui seront programmés dans le logiciel, je dispose d'un langage de script très puissant (utilisé dans Cogui) basé sur la réflexivité du langage Java. On peut envisager de l'utiliser à la fois pour calculer des attributs à partir du texte d'un élément de corpus et de ses attributs natifs mais aussi pour définir de nouvelle comparaisons entre attributs. Mais cela reste dans le domaine des contraintes que vous qualifiez de simple. Un problème des graphes conceptuels est que le bout de logique sur lequel ils sont fondés ne gère pas le OU. Pour l'instant, il me parait difficile d'utiliser le mécanisme SI ALORS SINON. En tous cas pas directement sous la forme idéale des règles avec défauts. Je sais que vous avez travaillé sur cette logique. J'ai récemment implémenté ce type de règle avec défaut qui permet de rechercher toutes les extensions possibles d'un ensemble de règle sur un fait. Ça marche et d'ailleurs un papier vient d'être accepté à l'ICCS (la conf. des GC) sur cette implémentation. Mais la complexité et surtout le fait qu'il n'y a pas d'optimisation permettant de limiter le nombre de réponses identiques rend le procédé très couteux. Mais on dispose d'un mécanisme de règle qui pourrait nous suffire : à chaque règle (classique) est adossé un bout de code (un script ou bien du code compilé) qui permet non seulement d'accepter/refuser qu'un règle soit appliqué mais aussi d'agir sur la conclusion de la règle. En outre je pense qu'il faudra réaliser la production par étape, le fait de fixer un élément choisi dans un corpus pourrait entrainer l'apparition d'attributs supplémentaires et déclencher de nouvelles règles. Ce procédé qui nous est imposé par certaines règles (par exemple le nombre de pieds d'un vers ne peut être calculé qu'après avoir appliqué les règles d'élision). Mais il est aussi un moyen de faire apparaitre des extensions différentes d'un même schéma de construction.

3.2.2.4. *Précisions sur les contraintes*

On doit distinguer les contraintes déjà imposées par la structure du schéma et celles qu'on y

ajoute. Prenons quelques exemples.
suite d'éléments choisis dans les corpus :

normalement, donnée par le parcours (de Tarry) des feuilles de l'arborescence. On peut prévoir des priorités dans l'algorithme général pour les fils d'un nœud, sinon, l'auteur doit les préciser (comme des poids par exemple, d'où contraintes supplémentaires d'inégalité)

Exemples (on suppose que les priorités sont incluses):

Phrase. Contraintes supplémentaires : accords (égalité)

Scénario (voir ALP[1], p.328 : 2 ou 3 corpus) ; contraintes de "réalisme" et autres.

Poèmes. Corpus de vers. Une strophe est une suite de vers, un poème une suite de strophes. Contraintes supplémentaires : **rimes, nombre de pieds, etc**

"le tra du des" . Un corpus (alors poids avec contrainte de supériorité) ou deux corpus. Contraintes supplémentaires : accords.

Petites annonces. Plusieurs corpus. Contraintes supplémentaires : compatibilité de catégories à définir par l'auteur.

Aphorismes (voir ALP p.315) Plusieurs corpus. Contraintes de compatibilité.

Concaténation d'expressions.

Concaténer des hémistiches pour fabriquer de nouveaux alexandrins.

Concaténer des expressions connues en les liant avec une proposition (voir ALP p 259) Ex : Attention à la marche ou crève

Fabriquer des "mots-valises"

Contraintes liées.

La contrainte sur un élément dépend des contraintes déjà appliquées ou des données

Exemples

Boule de neige (Voir ALP p 194). **Poème dont chaque vers est fait d'un mot qui augmente d'une lettre à chaque vers.** Le Corpus doit contenir des mots dont le nombre de lettres est un attribut (il peut y avoir d'autres attributs !). La contrainte change à chaque vers. Si C(i) est : mot de i lettres, alors C(i+1) est : mot de i+1 lettres.

Acrostiche (Voir PLS[2] p.103) Poème dont chaque vers commence par une lettre (ou un mot) donnée.

Modifier un texte.

Pour pouvoir modifier un texte, il faut la possibilité de saisir ses éléments sur un schéma et ensuite d'en changer certains par tirage avec contraintes.

Exemples. Littérature itérative (Voir ALP p.85)

Remplacer des mots par leur définition, récursivement.

A priori la mécanique envisagée répond bien à certains des exemples, mais pas à tous. Le prototype nous permettra de mieux cerner ce qu'il manque et ajouter des fonctionnalités

[1] Atlas de Littérature Potentielle

[2] Pour la Science, nov 1998