Langage de définition et de négociation des contraintes pour la résolution des problèmes sur-contraints

Wassim Jaziri– Michel Mainguenaud

Laboratoire PSI, Université et INSA de Rouen
Place Emile blondel, 76821 Mont Saint Aignan Cedex, France
wassim.jaziri@insa-rouen.fr, michel.mainguenaud@insa-rouen.fr

RÉSUMÉ. *Les territoires agricoles sont constamment confrontés à des processus de ruissellement et d'inondations qui sont à l'origine de dégâts considérables sur les plantations et l'habitat. Pour minimiser les dégâts et apporter des solutions efficaces aux décideurs, il est indispensable de tenir compte de leurs contraintes et de les intégrer dans la prise de décision. Cependant, les décideurs expriment souvent des contraintes multiples et complexes, ce qui fait que les problèmes de gestion du risque sont généralement sur-contraints. Nous proposons dans cet article d'intégrer les décideurs dans la prise de décision par l'expression et la relaxation de leurs contraintes. Un langage de définition et de négociation des contraintes permet aux décideurs d'exprimer et de mettre à jour leurs contraintes à travers une négociation avec le système.*

ABSTRACT. *Agricultural territories are constantly confronted to processes of streaming and floods that cause considerable damage on the plantations and on the habitat. To reduce damage and to provide efficient solutions to decision-makers, it is essential to take into account their constraints and to integrate them in the decision process. However, decision-makers often require multiple and complex constraints that make the problem over-constrained. We propose in this paper to integrate decision-makers in the decision process by the expression and the relaxation of their constraints. A language of constraints definition and negotiation allows decision-makers to express their constraints and to update them through a negotiation with the system.*

MOTS-CLÉS : *Gestion des risques de ruissellement, Problème sur-contraint, Définition et manipulation des contraintes, Négociation.*

KEYWORDS: *Management of streaming risks, Over-constrained problem, Constraints definition and manipulation, Negotiation.*

SAGEO'2005, pages 1 à 13

1. Introduction

Les processus de ruissellement, d'érosion et d'inondations font peser des risques importants sur les territoires agricoles (Langlois et al., 2002) et motivent notre étude. Différentes réflexions ont été engagées pour comprendre, analyser et proposer des solutions de lutte contre ces risques, entre autres dans le contexte agronomique (Boiffin et al. 1988) (Celik et al., 1996) (Gallien et al., 1995) (Hauchard et al., 2002) (Martin et al., 1998) et informatique (Barreteau et al., 2000) (Chen et al., 1999) (D'Aquino et al., 2002) (Le Ber et al., 1998) (Manche et al., 2002).
Cependant, bien que les moyens d'actions sont généralement entre les mains des décideurs (les exploitants), ces derniers sont rarement intégrés dans la prise de décision. Le concepteur simule souvent les choix et comportements des décideurs sans les intégrer effectivement dans le processus de résolution. Dans la plupart des cas, les décideurs accueillent de manière négative les solutions proposées et souhaitent ne pas être exclus de la concertation et des éventuelles prises de décision. Il est ainsi impératif de les intégrer dans la prise de décision et de les aider à décider au lieu de décider à leur place.

Par ailleurs, les décideurs sont souvent à l'origine de contraintes complexes et difficiles à satisfaire simultanément. Dans la plupart des travaux de satisfaction des contraintes (Tsang, 1993), la résolution de l'aspect sur-contraint se fait indépendamment des décideurs et dépend du point de vue et des compétences du concepteur de l'application. La relaxation des contraintes et les extensions du formalisme de satisfaction des contraintes ont fait l'objet de nombreuses réflexions (Fargier et al., 1993) (Freuder et al., 1992) (Schiex, 1992) (Tanguiane, 1991). Cependant, les décideurs sont peu impliqués dans ces travaux, voire absents du processus de prise de décision. Leur seule implication ressentie dans certains travaux, se limite à l'expression de leurs préférences ou leurs points de vue vis-à-vis des solutions proposées. Nous pensons que les décideurs restent les plus concernés par la relaxation de leurs contraintes et qu'il est nécessaire de leur donner les moyens pour les définir et les mettre à jour.
Après une présentation du cadre de notre travail, nous proposons un langage de définition et de négociation des contraintes des décideurs. Un ensemble d'opérations primitives, complexes et dérivées sont définies pour permettre aux décideurs de négocier avec le système et d'exprimer leurs choix et exigences.

2. Cadre de notre travail

Dans le cadre d'une action d'aide à la décision, l'expression d'un problème peut prendre deux aspects principaux. Le premier consiste à spécifier un problème de manière sous-contrainte provoquant ainsi une mise à disposition d'un ensemble

important de solutions. Le deuxième consiste à spécifier un problème de manière sur-contrainte entraînant par là une absence de solution. Nous nous plaçons dans ce deuxième cadre en supposant que le nombre de contraintes exprimées par les décideurs est important.
Face à un ensemble de contraintes, le problème le plus classique est de déterminer si cet ensemble de contraintes est cohérent. Comme ce problème est largement abordé dans la littérature, nous considérons que notre ensemble de contraintes est cohérent. Partant de ce postulat de cohérence, le problème suivant consiste à déterminer si le problème auquel nous sommes confrontés dispose d'une solution. Statistiquement, l'instrumentation de la résolution d'un problème sur-contraint n'est pas pertinente, puisque sans solution par définition. Notre problème formel final consiste donc dans un premier temps à offrir un environnement de spécification et de définition des contraintes puis dans un deuxième temps, de définir et de mettre à la disposition des décideurs des opérations de manipulation permettant de passer d'un problème sur-contraint à un problème sous-contraint.

3. Environnement de spécification et de définition des contraintes

Nous envisageons d'accorder aux décideurs la possibilité d'exprimer directement leurs choix et exigences suivant une syntaxe bien définie. Ceci est réalisable formellement au moyen d'un langage défini à l'aide d'une grammaire[1] générative. L'interface homme/machine a fait l'objet de nombreux travaux (Cole et al., 2000) (Weng, 2000). Nous nous positionnons ici au niveau conceptuel/algorithmique.
Les contraintes exprimées par les décideurs sont classées suivant un ordre de priorité traduisant leur pertinence à l'application afin de favoriser la satisfaction des plus pertinentes. La pertinence est établie en fonction de trois paramètres : utilisateur, système et métier. La liste ordonnée des contraintes reflète le plan de satisfaction (l'ordre de satisfaction des contraintes par le système). Les contraintes sont ensuite négociées entre le système et les décideurs pour retenir un sous-ensemble de contraintes à prendre en compte effectivement lors de la résolution. Les contraintes retenues en accord avec les décideurs représentent le graphe opérationnel de satisfaction (dépendant de l'application).

3.1. Classement des contraintes

Les contraintes sont représentées chacune sous forme d'un regroupement de cinq paramètres : l'identifiant de la contrainte, sa spécification textuelle, sa valeur de préférence, sa complexité et sa pertinence métier. Ces trois derniers paramètres

1. Une grammaire formelle est formée par un vocabulaire terminal, un ensemble de variables, un axiome et un ensemble de règles de productions.

représentent les paramètres utilisateur, système et métier utilisées pour ordonner les contraintes dans le plan de satisfaction.

- **Paramètre Utilisateur** relatif au **N**iveau de **P**référence (NP) des contraintes. Ce paramètre reflète l'importance de chaque contrainte pour le décideur et la nécessité de la satisfaire par le système. Il permet ainsi de classer les contraintes par niveau de préférence afin de privilégier la satisfaction des contraintes les plus préférées des décideurs. En effet, dans les applications réelles, les contraintes à satisfaire n'ont pas généralement la même importance. Les travaux autour des problèmes de satisfaction des contraintes distinguent notamment les contraintes dures qu'il faut absolument satisfaire, des contraintes souples qu'il est préférable de satisfaire (Schiex, 1992). Dans notre contexte, nous introduisons un troisième niveau de préférence. Nous considérons que les décideurs exigent à la fois des contraintes *dures*, qui traduisent des obligations et doivent impérativement être satisfaites, des contraintes *souples ou flexibles*, qui représentent des préférences et sont à satisfaire au mieux et des contraintes *secondaires*, dont l'insatisfaction contraint peu la qualité (pour le décideur) de la solution proposée. Le système favorise la satisfaction des contraintes dures puis flexibles. Les contraintes secondaires quant à elles, bien que peu importantes pour le décideur, peuvent servir de critère supplémentaire de décision pour le choix de la solution finale.
- **Paramètre Système** relatif au niveau de complexité des contraintes. Ce paramètre permet de classer les contraintes par ordre croissant de leur complexité temporelle afin de traiter en premier les moins coûteuses en temps. La complexité temporelle est une évaluation du temps nécessaire à la satisfaction d'une contrainte. Dans notre cas, nous exprimons la complexité d'une contrainte en fonction du nombre d'entités qu'elle implique et du nombre d'instances de chacune. Pour chaque entité impliquée dont la clé n'est pas donnée, la complexité augmente avec le nombre d'instances de l'entité concernée (Jaziri, 2004).
- **Paramètre Métier** relatif à la pertinence métier des contraintes. Un coefficient est attribué à chaque contrainte et représente sa pertinence métier par rapport à l'application, indépendamment de son niveau de préférence pour le décideur. Exemple, dans un contexte de gestion de risque, les contraintes hydrologiques ont un coefficient plus important que les contraintes spatiales.

3.2. Représentation des contraintes

Pour représenter les contraintes des décideurs et les paramètres utilisateur, système et métier, nous utilisons un formalisme de type abstrait de données TAD (Wirth, 1986). Un TAD est défini à partir des domaines de base, à savoir : Entier, Réel, Chaîne de caractères, Booléen, Temps et de constructeurs d'agrégation [ ], d'ensemble { }, de liste <> et d'énumération de constantes de type ().
Nous considérons une structure `contraintesApplicatives` (figure 1), reflétant le plan de satisfaction des contraintes décideur, et qui représente une agrégation

(induite du paramètre utilisateur) de trois Niveaux de Préférences (NP) relatifs respectivement aux contraintes dures, flexibles et secondaires :

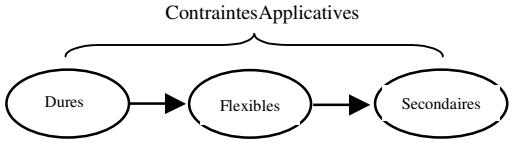

Figure 1. *Représentation de la structure* `contraintesApplicatives`*.*
Les flèches indiquent le sens du passage du NP prioritaire au moins prioritaire.

```
Type contraintesApplicatives =
[
   Dures : NP,
   Flexibles : NP,
   Secondaires : NP
]
```

Chaque `NP` (figure 2) représente une agrégation (induite du paramètre système) de trois niveaux de complexité, relatifs à la complexité d'évaluation en nombre d'opérations vis à vis du nombre de données manipulées. Nous distinguons les niveaux de complexité constante, linéaire (complexité $1<\Phi\leq n$) et surélevée [2] ($\Phi>n$) :

```
Type NP =
[
   Constantes : niveauComplexité,
   Linéaires : niveauComplexité,
   Surélevées : niveauComplexité
]
```

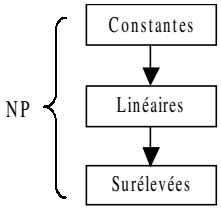

Figure 2. *Représentation de la structure* `NP`*.*

2. Regroupant les autres types de complexité non linéaire, entre autres : nlogn, n^2, n^3, etc.

La structure `contraintesApplicatives` représente donc une agrégation d'agrégation de trois niveaux de complexité.

Chaque niveau de complexité (`niveauComplexité`) est une collection non ordonnée (ensemble) de contraintes décideur. L'introduction du critère de pertinence métier (induit du paramètre métier) réordonne les contraintes à l'intérieur de chaque niveau de complexité de façon à avoir les contraintes les plus pertinentes en tête.
Chaque niveau de complexité représente ainsi une liste de contraintes ordonnées (par ordre décroissant) suivant leur pertinence métier :

`Type` *`niveauComplexité`* `= <contrainteDécideur >.`

Nous représentons formellement une `contrainteDécideur` (figure 3) par une agrégation de cinq composants, selon ce formalisme :

```
Type contrainteDécideur =
[
   identifiant : Chaîne de caractères,
   spécification : Chaîne de caractères,
   valeurPréférence : préférenceDécideur,
   valeurComplexité : complexitéConceptuelle,
   pertinence : Réel
]
```

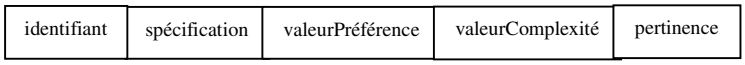

Figure 3. *Représentation de la structure* `contrainteDécideur`.

Les types `préférenceDécideur` et `complexitéConceptuelle` représentent des types énumérés définis comme suit :

`Type` *`préférenceDécideur`* `= (DURE, FLEXIBLE, SECONDAIRE)` avec `DURE`, `FLEXIBLE` et `SECONDAIRE` sont des constantes de type `préférenceDécideur`.

`Type` *`complexitéConceptuelle`* `= (CONSTANTE, LINEAIRE, SURELEVEE)` avec `CONSTANTE`, `LINEAIRE` et `SURELEVEE` sont des constantes de type `complexitéConceptuelle`.

Le plan de satisfaction (figure 4) ainsi obtenu peut être assimilé à une liste de contraintes ordonnées suivant la valeur de préférence, la valeur de complexité et la pertinence métier, et définir ainsi un ordre total :

`planSatisfaction` $= \langle\ c^{i}_{jk}\ \rangle$ avec c^{i}_{jk} : la contrainte appartenant au Niveau de Préférence *i*, de niveau de complexité *j* et de rang *k* (ordre décroissant suivant la pertinence métier) dans le niveau de complexité *j*.

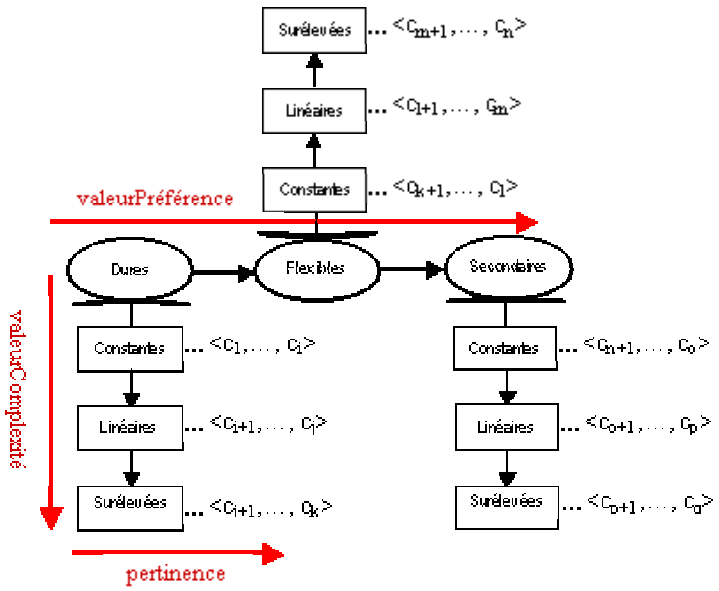

Figure 4. *Plan de satisfaction des contraintes.*

4. Manipulation et négociation des contraintes

Nous souhaitons sensibiliser les décideurs aux difficultés rencontrées par le concepteur pour satisfaire leurs contraintes. A cet effet, nous les intégrons dans une phase de négociation avec le système pour leur permettre de contrôler et de mettre à jour leurs contraintes. Cette négociation est basée sur le principe « négocier pour convaincre ». En effet, pour satisfaire l'ensemble des contraintes exigées par les décideurs, le système doit disposer des ressources suffisantes nécessaires à leur satisfaction. Ces ressources exprimées en terme de temps de réponse peuvent être déterminées à l'aide du paramètre de complexité qui révèle le coût nécessaire pour la satisfaction de chaque contrainte et reflète donc la complexité temporelle de l'application.

4.1. Déroulement de la négociation

Le décideur et le système mènent une négociation ayant pour finalité de trouver un compromis équilibrant le rapport : temps de réponse nécessaire (coût total des contraintes) / temps de réponse accordé par les décideurs (ressources système).

L'équilibre de ce rapport de coût peut se faire en modifiant le système de contraintes et/ou le temps de réponse autorisé pour fournir une solution.
La négociation est guidée par le système et gérée par le décideur. Ce dernier intervient au début du processus pour exprimer ses besoins sous forme de contraintes et ses préférences sur ces besoins. A partir de cette première série de contraintes introduites par les décideurs, le système évalue la complexité temporelle des contraintes et visualise ce coût au décideur. Ce dernier, en fonction des coûts relatifs aux contraintes qu'il exige et du temps d'attente qu'il autorise au système avant de fournir une solution, tente d'équilibrer le rapport entre ces deux paramètres. Un ensemble d'opérations sur le système de contraintes est ainsi défini afin de permettre aux décideurs de réduire le nombre de contraintes à satisfaire ou d'augmenter le temps de réponse autorisé. Nous faisons ici abstraction du processus de saisie et de visualisation des contraintes des décideurs en s'intéressant plus particulièrement au processus de manipulation et de mise à jour de ces contraintes.

4.2. Opérations de manipulation des contraintes

Pour illustrer les opérations de manipulation des contraintes, nous nous basons sur les structures de référence : `contraintesApplicatives` reflétant le plan de satisfaction et `listeContraintes` représentant une liste d'éléments de type `contrainteDécideur` :

```
Type listeContraintes = <contrainteDécideur>.
```

Nous utilisons dans ce qui suit les notations suivantes :

→ : Retour de l'opération ;

× : Séparateur des paramètres de l'opération.

4.2.1 Opérations primitives

Les principales opérations primitives sont les suivantes :

- ***Insérer :*** `listeContraintes × contrainteDécideur → listeContraintes.`

L'opération `Insérer` ajoute une contrainte décideur à la liste initiale suivant sa valeur de préférence, sa valeur de complexité et sa pertinence. Elle retourne une liste de longueur supérieure à la liste initiale.

- ***Supprimer :*** `listeContraintes × contrainteDécideur → listeContraintes.`

L'opération `Supprimer` retire une contrainte décideur de la liste initiale. Elle retourne une liste de longueur inférieure à la liste initiale.

- ***Modifier :*** `contrainteDécideur × préférenceDécideur → contrainteDécideur.`

L'opération `Modifier` met à jour la valeur de préférence d'une contrainte. Elle retourne une contrainte de même valeur de complexité et pertinence métier.

- ***Sélectionner :*** `listeContraintes × conditionSélection → listeContraintes.`

L'opération `Sélectionner` permet d'extraire une liste de contraintes vérifiant une condition de sélection. Elle retourne une liste, éventuellement vide, de contraintes de longueur inférieure ou égale à la liste initiale.

4.2.2 Opérations complexes

Les principales opérations complexes à la disposition du décideur sont :

- ***Extraire:*** `contraintesApplicatives × préférenceDécideur→ NP.`

L'opération polymorphe[3] `Extraire` permet d'extraire un niveau de préférence NP de la structure contraintesApplicatives.

Exemple : Extraire les contraintes secondaires (figure 5) :

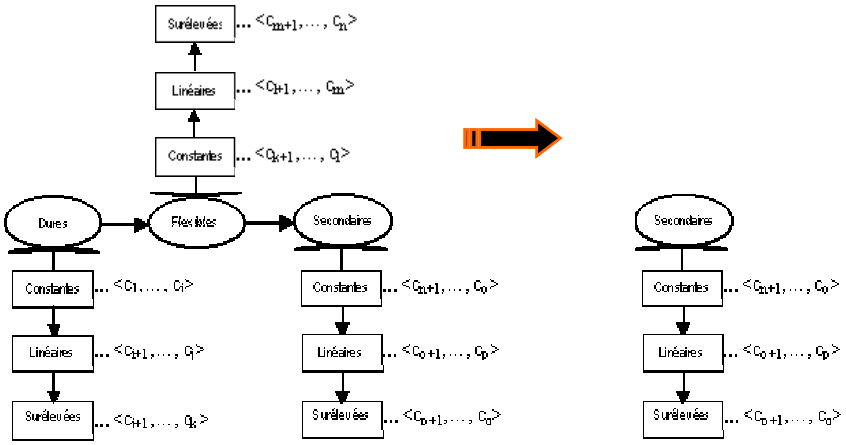

Figure 5. *Illustration de l'opération* `Extraire`.

- ***Extraire :*** `NP × complexitéConceptuelle → niveauComplexité.`

L'opération polymorphe `Extraire` permet d'extraire un niveau de complexité (correspondant à la complexité conceptuelle spécifiée) relatif à un NP donné.

- ***Tailler :*** `contraintesApplicatives × préférenceDécideur → contraintesApplicatives.`

3. Une opération est dite polymorphe si elle accepte des arguments de types différents.

L'opération polymorphe `Tailler` revient à supprimer un niveau de préférence de la structure contraintesApplicatives, i.e., supprimer les contraintes de valeur de préférence égale à préférenceDécideur. Cette opération retourne une structure, éventuellement vide, de contraintes, incluse dans la structure initiale.

- ***Tailler :*** `NP × complexitéConceptuelle → NP.`

L'opération polymorphe `Tailler` revient à supprimer un niveau de complexité appartenant à un NP donné. Cette opération retourne un niveau de préférence NP, éventuellement vide, inclus dans le NP initial.

- ***Tailler :*** `contraintesApplicatives × complexitéConceptuelle → contraintesApplicatives.`

L'opération polymorphe `Tailler` revient à supprimer un niveau de complexité de la structure contraintesApplicatives, i.e., supprimer les contraintes de valeur de complexité égale à complexitéConceptuelle. Cette opération retourne une structure contraintesApplicatives, éventuellement vide, incluse dans la structure contraintesApplicatives initiale.

- ***Tailler :*** `contraintesApplicatives × listeContraintes → contraintesApplicatives.`

L'opération polymorphe `Tailler` revient à supprimer une liste de contraintes de la structure contraintesApplicatives. Cette opération retourne une structure contraintesApplicatives, éventuellement vide, incluse dans la structure de contraintes initiale. Elle n'est pas définie si les contraintes à tailler n'appartiennent pas à la structure contraintesApplicatives.

- ***Insérer :*** `contraintesApplicatives × listeContraintes → contraintesApplicatives.`

L'opération `Insérer` ajoute une liste de contraintes décideur à la structure contraintesApplicatives. Les contraintes ajoutées seront classées suivant leur valeur de préférence, leur valeur de complexité et leur pertinence métier. Cette opération retourne une structure contraintesApplicatives de cardinalité (de contraintes) supérieure à la structure initiale.

- ***Modifier :*** `listeContraintes × préférenceDécideur → listeContraintes.`

L'opération `Modifier` met à jour la valeur de préférence d'une liste de contraintes tout en gardant la même valeur de complexité et pertinence métier. Elle retourne une liste de contraintes de même longueur que la liste initiale.

- ***Evaluer :*** `contraintesApplicatives → Temps.`

L'opération `Evaluer` mesure le temps nécessaire pour satisfaire toutes les contraintes de la structure contraintesApplicatives. Cette opération retourne la complexité temporelle globale des contraintes de la structure.

- ***Respecter :*** `contraintesApplicatives × Temps → Booléen.`

L'opération `Respecter` revient à ne pas dépasser un temps maximum autorisé pour fournir une solution satisfaisant les contraintes de la structure

contraintesApplicatives. Cette opération retourne un booléen relatif à l'état de satisfaction du temps exigé.

4.2.3 Opérations dérivées

Nous définissons l'opération dérivée `Replacer`, permettant la réorganisation des contraintes suivant une nouvelle valeur de préférence :

- ***Replacer :*** `contraintesApplicatives × préférenceDécideur × complexitéConceptuelle × préférenceDécideur →` `contraintesApplicatives.`

L'opération `Replacer` revient à modifier, dans une structure contraintesApplicatives, le positionnement des contraintes vérifiant une préférenceDécideur et une complexitéConceptuelle données. L'opération `Replacer` retourne une structure contraintesApplicatives de même cardinalité (nombre de contraintes) que la structure initiale.

Exemple : Replacer au Niveau de Préférence Secondaires, les contraintes de complexité surélevée et de valeur de préférence dure (figure 6) :

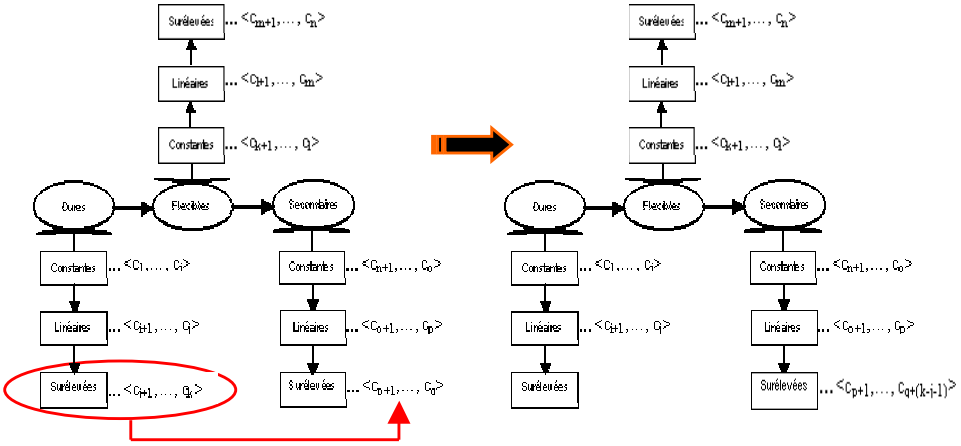

Figure 6. *Illustration de l'opération* `Replacer`.

La phase de négociation permet d'établir, à partir du plan de satisfaction, le graphe opérationnel des contraintes à satisfaire par le système. Les opérations appliquées par le décideur sont contrôlées par le système afin de visualiser les conséquences des modifications (en terme de coût) apportées sur la liste des contraintes. Les choix et opérations effectués par les décideurs déterminent la marge de manœuvre accordée au concepteur et influent sur la qualité de la solution pouvant être proposée par le système.

5. Conclusion et perspectives

Nous avons présenté dans cet article une méthodologie de construction et de négociation des contraintes des décideurs, basée sur un langage fonctionnel de manipulation des contraintes. Les décideurs sont les acteurs du système au travers de la définition et la mise à jour de leurs contraintes au moyen d'opérations primitives, complexes et dérivées. L'avantage de notre approche est sa souplesse de manipulation au niveau informatique mais bien évidemment elle ne représente pas le niveau d'interaction souhaité avec des acteurs non familiers de l'outil informatique. En effet, le langage proposé peut servir de support, à travers une interface homme-machine, pour mener une négociation entre des décideurs et le système. Néanmoins, pour être opérationnel, il conviendrait d'exprimer les contraintes de manière conviviale. L'aspect interface homme-machine représente donc un axe de recherche pour permettre d'exprimer facilement des contraintes.

6. Bibliographie

Barreteau O., Bousquet F., SHADOC: a multi-agent model to tackle viability of irrigated systems, *Annals of Operations Research*, volume 94, p. 139-162, 2000.

Boiffin J., Papy F., Eimberck M., Influence des systèmes de culture sur les risques d'érosion, *Agronomie*, p. 663-673, 1988.

Celik I., Aydin M., Yazici U., A review of the erosion control studies during the republic period in Turkey, *Proceedings of 1st International Conference on Land Degradation*, p. 175-180, Turquie, 1996.

Chen F., Kanemasu E.T., West L.T., Rachidi F., Analysis of land use and simulation of soil erosion with GIS for the semi-arid region of Morocco, *Géo-observateur*, p. 55-75, 1999.

Cole M., O'Keefe R., Siala H., From the user interface to the consumer interface, *Information Systems Frontiers*, volume 1 (4), p. 349-361, Kluwer Academic Publishers, 2000.

D'Aquino P., Le Page C., Bousquet F., Bah A., Une expérience de conception directe de SIG et de SMA par les acteurs dans la vallée du Sénégal, *Géomatique*, volume 12, p. 517-542, 2002.

Fargier H., Lang J., Schiex T., Selecting preferred solutions in fuzzy constraint satisfaction problems, *Proceedings of the 1st European Congress on Fuzzy and Intelligent Technologies (EUFIT)*, p. 128-134, Allemagne, 1993.

Freuder E.C., Wallace R.J., Partial constraint satisfaction, *Artificial Intelligence*, p. 21-70, 1992.

Gallien E., Le Bissonnais Y., Eimberck M., Benkhadra H., Ligneau L., Ouvry J.F., Martin P., Influence des couverts végétaux de jachère sur le ruissellement et l'érosion diffuse en sol limoneux cultivé, *Cahiers Agricultures*, p. 171-183, 1995.

Hauchard E., Delahaye D., Freire-Diaz S., Organisation fractale de l'occupation du sol :

conséquences sur le ruissellement et le ravinement dans les terres de grande culture. *Géomorphologie, relief, processus, environnement*, p. 181-196, 2002.

Jaziri W., Modélisation et gestion des contraintes pour un problème d'optimisation sur-contraint : application à l'aide à la décision pour la gestion du risque de ruissellement, Thèse de Doctorat en Informatique, INSA de Rouen, 9 juillet 2004 : http://tel.ccsd.cnrs.fr/documents/archives0/00/00/81/24/index_fr.html.

Langlois P., Delahaye D., RuiCells, Automate cellulaire pour la simulation du ruissellement de surface, *Géomatique*, volume 12, n° 4, p. 461-487, 2002.

Le Ber F., Chevrier V., Dury A., A multi-agent system for the simulation of land use organisation, *Proceedings of 3rd IFAC/CIGR Workshop on Artificial Intelligence in Agriculture*, p. 182-187, Japon, 1998.

Manche Y., Villanova M., Martin H., Burnet R., Un système d'information dans le domaine des risques naturels : le projet SIRVA, *Géomatique*, volume 12, p. 59-76, 2002.

Martin P., Papy F., Souchère V., Capillon A., Maîtrise du ruissellement et modélisation des pratiques de production, *Cahiers Agricultures*, p. 111-119, 1998.

Schiex T., Possibilistic constraint satisfaction problems or "How to handle soft constraints?", *Proceedings of the 8th International Conference on Uncertainty in Artificial Intelligence* (UAI-92), p. 268-275, USA, 1992.

Tanguiane A.S., *Aggregation and representation of preferences*, Springer-Verlag, Berlin, 1991.

Tsang E.P.K., *Foundations of constraint satisfaction*, Academic Press, London, 1993.

Weng O., Human-environment interactions in agricultural land use in a South China's wetland region: a study on the Zhujiang Delta in the Holocene, *GeoJournal*, volume 51, p. 191-202, Kluwer Academic Publishers, 2000.

Wirth N., *Algorithms and data structures*, Prentice-Hall, USA, 1986.