HAL open science

Modélisation par contraintes et exploitation d'une gamme automobile : nouveaux problèmes, nouvelles requêtes, nouveaux besoins en programmation par contraintes

Jeam-Marc Astesana, Laurent Cosserat, Hélène Fargier

To cite this version:

Jeam-Marc Astesana, Laurent Cosserat, Hélène Fargier. Modélisation par contraintes et exploitation d'une gamme automobile : nouveaux problèmes, nouvelles requêtes, nouveaux besoins en programmation par contraintes. 6èmes Journées Francophones de Programmation par Contraintes (JFPC 2010), Groupe de Recherche en Informatique, Image, Automatique et Instrumentation de Caen (GREYC), Jun 2010, Caen, France. pp.33-42. inria-00520306

HAL Id: inria-00520306

https://inria.hal.science/inria-00520306

Submitted on 22 Sep 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Modélisation par contraintes et exploitation d'une gamme automobile : nouveaux problèmes, nouvelles requêtes, nouveaux besoins en programmation par contraintes

Jean Marc Astesana[1] **Laurent Cosserat**[1] **Hélène Fargier**[2]

[1] Renault SA, 13 avenue Paul Langevin 92359 Plessis Robinson
[2] IRIT-CNRS, 118 route de Narbonne 31062 Toulouse Cedex 9
{jean-marc.astesana,laurent.cosserat}@renault.com fargier@irit.fr

Abstract

La mise au point et l'exploitation de la gamme commerciale d'un constructeur automobile présentent de multiples problèmes métier qui peuvent se modéliser comme des problèmes CSP : détection et analyse d'incohérence dans la spécification de la gamme, configuration véhicule, configuration de la prévision. Ces problèmes mettent en jeu des données de taille importante, et leur résolution en situation réelle doit être effectuée dans un temps de calcul limité. Cet article présente, formalise et étudie la complexité de plusieurs de ces problèmes. Au-delà de l'étude d'une application réelle, son objectif est de proposer à la communauté CSP non pas de nouvelles approches de résolution, mais de nouvelles requêtes, de nouveaux problèmes, voire de nouveaux besoins.

1 Introduction

Le formalisme des problèmes de satisfaction de contraintes (CSP) offre un cadre de formalisation puissant pour l'expression d'une multitude de problèmes, en particulier de nombreux problèmes académiques ou issus du monde réel (par exemple les problèmes de coloration de graphes, d'ordonnancement, d'affectation de fréquence, . . .). Résoudre un CSP c'est, quasiment toujours, déterminer s'il est cohérent et/ou lui trouver une solution -dans certains cas une solution optimale- et c'est une question difficile : le problème de décision associé est NP-complet. La grande majorité des travaux de la communauté a donc porté sur ce problème, soit en l'abordant de front (proposer les algorithmes les plus efficaces possibles pour déterminer la cohérence d'un CSP), soit en cherchant comment le contourner, par l'étude de sous-classes qui seraient polynomiales ou par la définition d'algorithmes sains mais incomplets.

Or si les CSP, ou les langages de programmation par contraintes, sont largement utilisés, ce n'est pas seulement parce qu'ils sont munis d'algorithmes de résolution efficaces. C'est aussi parce qu'ils permettent d'exprimer simplement les données des applications modélisées. Mais dans ces applications, la question n'est pas toujours de savoir si le CSP est cohérent ni même d'en trouver une solution. Les applications en configuration de produit, par exemple, définissent des CSP cohérents (pourvus de centaines de milliers de solutions) ... et la machine n'est pas utilisée pour extraire une solution, mais pour aider l'utilisateur à construire la sienne.

L'ambition de cet exposé est de proposer à la communauté non pas de nouvelles approches de résolution, mais de nouvelles requêtes, de nouveaux problèmes, voire de nouveaux besoin en modélisation, en exposant quelques problèmes métier de Renault qui ont pour point commun la nécessité d'inférer sur le système de contraintes représentant la gamme, et notamment la configuration de véhicules automobiles.

2 Modélisation à base de contraintes de la gamme Renault

La gamme de véhicules de Renault est très étendue. Le nombre de véhicules différents dans la gamme d'un modèle peut être énorme. Par exemple, l'ensemble des petites fourgonnettes "Trafic" comprend 10^{21} véhicules différents. Ce nombre rend impossible l'énumération de l'ensemble de ces véhicules dans une base

de données. Cet ensemble doit alors être représenté en intention. La gamme peut être vue comme spécifiée par modèle, et dans la suite de cet article on entend par "gamme" l'ensemble de toutes les différentes variantes possibles d'un modèle. Les véhicules sont définis par des variables ($carburant$, $couleur$, etc.) qui prennent des valeurs dans des domaines énumérés ($\{essence, diesel, gpl\}$, $\{blanc, noir, rouge\}$, etc.). Les choix possibles pour ces variables ne sont pas indépendants les uns des autres (typiquement, un niveau d'équipement donné n'est pas disponible sur toutes les motorisations) : des dépendances sont modélisées par un système de contraintes entre ces variables. On a ainsi tous les ingrédients d'un problème CSP, dont chaque solution définit un véhicule précis de la gamme.

Plusieurs travaux ont proposé d'utiliser des extensions des CSP pour formaliser des problèmes de configuration, traitant généralement de difficultés de représentation spécifiques à ces types de problèmes, comme par exemple le fait que l'existence d'une variable puisse dépendre la valeur affectée à une autre. L'étude des problèmes de configuration a ainsi poussé la communauté à étendre le modèle CSP de base, définissant ainsi les CSP dynamiques [8], CSP composites [10], CSP interactifs [7], CSP à hypothèses [1], etc. Dans cet article, nous négligerons la prise en compte de ces difficultés de représentation pour nous tourner plutôt vers les requêtes adressées au modèle, vers la question de l'exploitation des données. *On supposera donc que la gamme est représentée par un CSP "classique".*

Les notations et définitions précises nécessaires à la formalisation de l'ensemble des questions étudiées ici seront introduites en Section 4. Disons pour l'instant simplement qu'un CSP est un triplet $P =< X, D, C >$ où X est l'ensemble de ses variables, D l'ensemble des domaines respectifs de ces variables et C l'ensemble (supposé fini) de ses contraintes ; on désignera par $Sols(P)$ l'ensemble des solutions de P.

Comme la plupart des instances issues de problèmes réels, les CSP représentant les modèles de la gamme Renault possèdent des spécificités particulières, en termes sémantiques (type de variables et contraintes) et syntaxiques (forme du graphe de contraintes).

La gamme de Renault est construite suivant ce qu'on appelle le "modèle version-options". Selon ce modèle, les variables se répartissent en deux catégories :

- un petit nombre de variables servant à définir les versions : appelons-les variables majeures. Soit $Maj \subset X$ leur ensemble : pour fixer les idées, on peut considérer que la restriction de $Sols(P)$ aux variables de Maj définit l'ensemble des versions de la gamme. Une variable encodant précisément la version peut exister, mais ce n'est pas systématiquement le cas : dans le cas contraire, la notion de version n'existe qu'implicitement, comme combinaison de valeurs de variables majeures.
- le reste des variables, la grande majorité, dont les valeurs possibles dépendent de la version : ce sont les variables mineures éléments de Min. Maj et Min forment une partition de X. Une valeur du domaine d'une variable mineure pourra ainsi être impossible sur une version, optionnelle sur une autre et obligatoire sur une troisième.

On peut regrouper comme suit les contraintes de spécification de la gamme :

- Les *contraintes majeures* portent sur des variables majeures qui définissent l'ensemble des versions. Typiquement, une contrainte majeure est directement définie par une Table, c'est-à-dire par l'énumération explicite des combinaisons autorisées de valeurs des variables de la contrainte.
- Les *contraintes options* sont des contraintes de la forme "$(x = v) \Rightarrow (y \in A)$", x étant une variable majeure, v une valeur de son domaine, y une variable mineure, et A un sous-ensemble du domaine de x. Par exemple, on peut utiliser ce type de contrainte pour spécifier la liste des autoradios disponibles sur les versions haut de gamme.
- Les *contraintes packs.* Il arrive qu'un lot de plusieurs options soit regroupé en une seule : un pack (ces options s'appellent alors les composants du pack). Les packs donnent lieu à de nombreuses contraintes qui complexifient le CSP. Par exemple, la contrainte de définition d'un pack exprime que le choix d'un pack *pack* implique la sélection de la totalité de ses n composants opt_i : $(x = pack) \Rightarrow \wedge_{i=1}^{n}(x_i = opt_i)$. De plus, une même option peut être un composant de plusieurs packs, qui s'excluent alors mutuellement.
- Les *contraintes quelconques.* Il reste des contraintes ne faisant pas partie des catégories précédentes. Elles peuvent inclure des variables majeures et mineures.

Notons que certaine contraintes, comme les contraintes options, s'expriment par des combinaisons booléennes de conditions ($variable = valeur$) ou de conditions ($variable \in$ sous-domaine) ; par exemple la contrainte $(carburant = diesel) \wedge (type_boite = boite_auto) \Rightarrow (type_chauffage = clim_regulee)$. On parle alors d'expression booléenne[1].

Dans les paramètres les plus dimensionnants des

[1]Il est bien entendu possible de modéliser la gamme de manière entièrement booléenne et d'obtenir ainsi des formules de la logique propositionnelle ; plusieurs codages sont possibles - voir par ex [13], mais pour des raisons de lisibilité, c'est souvent l'encodage direct qui est utilisé : pour chaque variable var qui admet les valeurs possibles $val_1, ..., val_n$ on définit n variables booléennes $val_1, ..., val_n$. Ces variables sont classiquement liées par des contraintes d'exhaustivité et d'exclusivité.

problèmes CSP définissant une gamme, on peut citer :

- La combinatoire des variables majeures. En configuration commerciale il y a au plus une centaine de versions, mais sur des gammes dites techniques, la projection de $Sols(P)$ sur les variables majeures peut en comporter plusieurs dizaines de milliers.
- le nombre de variables mineures et majeures. On peut supposer que $card(Min) \leq 150$. Le nombre de variables majeures est de l'ordre de 10.

La complexité est aggravée par les packs : au pire on peut en avoir une dizaine, chacun impliquant typiquement de 2 à 5 options composantes.

Mais l'essentiel de la difficulté vient des contraintes quelconques. Contrairement aux contraintes options, qui relient les variables de manière essentiellement hiérarchique, ces contraintes peuvent porter sur n'importe quelles variables, et donc complexifier le graphe de contraintes. Il peut y en avoir jusqu'à une centaine : elles portent généralement sur moins de quatre variables (sur deux variables dans la moitié des cas).

3 Exploitation de la gamme

Nous décrivons maintenant les requêtes métier qui se posent chez Renault lors des tâches d'élaboration, la maintenance et l'exploitation d'une gamme de véhicules - dit autrement, lors de la définition, la modification ou l'exploitation de ce CSP. Leur formalisation en termes de CSP sera abordée en Section 4.

3.1 Élaboration de la documentation véhicule

Les premières tâches interviennent en amont du problème de configuration client, en phase de modélisation de la gamme par un système de contraintes. On parle d'élaboration de la "documentation véhicule". Il s'agit de savoir si la gamme décrite par le système est cohérente, dans un sens étendu : dans certains cas, il ne suffit pas qu'il existe des véhicules compatibles avec les contraintes, il faut aussi que les possibilités spécifiées par les contraintes soient effectives. Supposons par exemple qu'une contrainte signifie "Les véhicules ayant le moteur M3 peuvent avoir comme type de chauffage : climatisation manuelle ou climatisation régulée." Si d'autres contraintes rendent incompatible le moteur M3 et la climatisation manuelle, la documentation véhicule est considérée comme incohérente. Autrement dit, certaines contraintes de la documentation véhicule ont une double sémantique : négative (elles interdisent des combinaisons de valeurs), et positive (les combinaisons autorisées par une contrainte doivent être effectivement possibles) : nous appelons cette propriété cohérence positive. C'est le cas des contraintes majeures et des contraintes options. Les questions qui se posent sont :

Cohérence (contextuelle) de la gamme : Y a-t-il au moins un véhicule répondant à cette définition de gamme ? Étant donnée une liste de quelques variables dont on impose la valeur, y a-t-il dans la gamme au moins un véhicule compatible avec cette affectation ? Plus généralement, étant donnée une expression booléenne c portant sur des paires $(variable = valeur)$, y a-t-il dans la gamme au moins un véhicule vérifiant c ?

Lorsque la réponse à ces questions est non, c'est que la documentation véhicule est erronée dans son état courant. Il faut alors analyser l'incohérence, c'est-à-dire déterminer un jeu minimal de contraintes qui suffit à expliquer l'impossibilité des véhicules prévus :

Conflits : Trouver un ensemble de contraintes incohérent, qui soit minimal pour l'inclusion.

Conflits contextuels : Étant donnée une liste de variables dont la valeur est imposée, trouver un ensemble de contraintes incohérent avec ces affectations qui soit minimal pour l'inclusion.

Cohérence positive Pour chaque contrainte à sémantique positive, et chaque tuple qu'elle autorise, existe il un véhicule de la gamme qui réalise ce tuple. De la même façon, pour chaque variable et chaque valeur de son domaine, existe-t-il un véhicule qui affecte cette valeur à cette variable.

3.2 Restitution de la gamme

Lorsque la gamme a été définie de manière cohérente, on cherche à en extraire des données significatives techniquement en considérant des ensembles de variables $\{x_1, \ldots, x_k\}$ (généralement, $k \leq 6$), et générant automatiquement des descriptions synthétiques des combinaisons des valeurs possibles de ces variables, typiquement sous la forme d'expressions booléennes sur des propositions de type $variable = valeur$.

Validité de description synthétique : Tester si tous les véhicules de la gamme vérifient une description synthétique c donnée.

Simplification de description synthétique : Une description synthétique c donnée est elle minimale (peut-on, sans supprimer une variable de c sans compromettre sa validité ?)

Génération de description synthétique : Étant donné un ensemble de variables, quelles sont les combinaisons de valeurs pour ces variables qui sont autorisées par la gamme (idéalement, on voudrait générer cette projection sous la forme d'une description synthétique minimale) ?

On cherche enfin à dimensionner la diversité de la gamme, en termes de nombres de véhicules différents :

Comptage : Combien de véhicules la gamme

compte-t-elle ? Combien de véhicules ayant un ensemble donné de valeurs imposées la gamme compte-t-elle ? Combien de combinaisons de valeurs différentes sont autorisées pour un sous-ensemble de variables Y : typiquement, on peut vouloir dimensionner la "diversité technique", c'est-à-dire le nombre de véhicules existant dans la gamme, hors considérations de teinte , appellation commerciale et pays de commercialisation.

3.3 Élaboration de la documentation pièces

La documentation pièces de Renault spécifie les pièces montées sur les véhicules de la gamme considérée. Elle est distincte de la documentation véhicule, dont les variables représentent des prestations perçues par les clients et non des pièces. Le lien est assuré par ce qu'on appelle le "cas d'emploi" : le cas d'emploi d'une pièce p_i est une expression booléenne c_i portant sur des atomes $variable = valeur$ de la documentation véhicule - cette contrainte spécifie les véhicules sur laquelle la pièce est montée. Par exemple, une pièce se monte sur les véhicules qui ont un moteur essence et une boîte automatique mais pas de toit ouvrant.

Les pièces sont regroupées en "pièces génériques". Une pièce générique, par exemple l'autoradio, est une fonction réalisée par la pièce. Pour que la documentation pièces soit cohérente, il faut que chaque véhicule ait un autoradio (propriété d'exhaustivité), qu'aucun véhicule n'en ait plusieurs (propriété de non-dualité) et que tout autoradio soit utilisé dans au moins un véhicule. On peut voir une pièce générique comme une variable supplémentaire p dont les valeurs sont les pièces associées $p_1, ..., p_n$. A chaque p_i correspond une expression booléenne c_i définissant son cas d'emploi. Les questions qui se posent alors sont :

Concision de la doc. : Pour chaque pièce associée (chaque cas d'emploi), existe-t-il au moins un véhicule de la gamme qui le valide ?

Exhaustivité de la doc. : Existe-t-il un véhicule ne montant aucune des pièces associées à une pièce générique donnée (ne validant aucun des cas d'emploi de la pièce générique) ?

Non-dualité des Emplois : Existe-t-il un véhicule montant deux des pièces, ou plus, associées à une pièce générique donnée (validant au moins deux des cas d'emplois de la pièce générique) ?

Si la réponse est "oui" à l'un des deux derniers points, on veut caractériser ces véhicules ou ces pièces :

Emplois Inconnus : Donner une condition sur les atomes "$variable = valeur$" qui caractérise l'ensemble des véhicules ne montant aucune des pièces associées à une pièce générique donnée ?

Emplois Multiples : Quelles sont les ensembles de pièces qui peuvent être montées simultanément sur un véhicule ; pour chaque ensemble, donner une condition sur les atomes "$variable = valeur$" qui caractérise l'ensemble des véhicules supportant ces pièces simultanément ?

Il peut sembler relativement simple de construire ces conditions caractéristiques en formant des combinaisons logiques à partir des cas d'emplois et des contraintes du CSP, si tant est qu'elles sont exprimées par des conditions logiques. La difficulté est que chaque condition caractéristique recherchée doit être simplifiée (aucune variable ne doit pouvoir en être supprimée sans compromettre l'ensemble qu'elle définit).

3.4 Configuration en ligne

Les problèmes liés à la configuration en ligne ont été déjà abordés dans plusieurs articles (voir par exemple [1]). Le principe de la configuration en ligne est que l'utilisateur (un client potentiel) définit le produit qu'il désire en choisissant interactivement des valeurs pour les variables ; il peut également revenir en arrière en relâchant un ou plusieurs de ses choix. Après chaque action, les domaines de valeurs possibles fournis à l'utilisateur doivent être modifiés de manière à contenir toutes les valeurs compatibles avec les choix courants, et seulement les valeurs compatibles avec ces choix. L'utilisateur peut forcer le choix d'une valeur détectée comme incompatible : le CSP devient alors incohérent. Quand l'ensemble des choix est incohérent avec les contraintes du CSP, il doit pouvoir revenir sur ses choix précédents et les relâcher.

L'utilisateur doit pouvoir également avoir une idée du prix du véhicule qu'il est en train de construire, et du délai d'obtention. Les informations prix sont décrites par des coûts associés aux valeurs de variables, ou par des fonctions de coût portant sur quelques variables. Le prix d'un véhicule est la somme des prix unitaires. Notons que certains coûts peuvent être négatifs (ce qui correspond à un retrait d'option par rapport au choix "standard"). On peut, en simplifiant, supposer que les informations délai sont portées par des conditions composées sur la gamme et définissant des intervalles de valeurs ("si le moteur choisit est hybride, alors le délai est d'au moins deux mois").

Pour aider l'utilisateur dans sa démarche, le système doit répondre aux requêtes suivantes :

Cohérence globale : Assurer, après chaque modification de domaine (choix de valeur ou relaxation) que les domaines correspondent exactement aux valeurs compatibles avec les choix courants.

Estimation du prix / du délai : Calculer le prix (ou le délai) min (ou max) des véhicules de la gamme compatibles avec les choix courants.

Calcul de restauration : En cas d'incohérence, identifier un ou des sous-ensembles maximaux

cohérents de restrictions utilisateur.

Calcul de conflit : Identifier des sous-ensembles minimaux de choix utilisateur incohérents, ou incohérents avec des choix de valeur de variable.

Simulation des choix : Déterminer quel serait l'état des domaines si l'utilisateur choisissait telle valeur pour telle variable (ce qui revient à la première requête, la cohérence globale)

Complétion de la configuration : Lorsqu'un certain nombre de variables "obligatoires" ont été renseignées, complétion automatique de la configuration courante à la demande de l'utilisateur (éventuellement sur un critère de minimisation de prix ou de délai).

3.5 Probabilisation de la gamme

La spécification de la gamme permet également à l'entreprise de construire des prévisions sur les ventes - typiquement, pour dimensionner sa production et celle de ses fournisseurs. Pour un modèle, ces prévisions sont équivalentes à la donnée d'un volume global et de probabilités (des taux prévisionnels) sur les valeurs de variables, conditionnées éventuellement par un contexte. ("On prévoit le choix de sièges en cuir dans 30% des véhicules vendus", "En Pologne, 25% des véhicules vendus seront diesel"). Ces probabilités individuelles sont en principe les projections d'une distribution de probabilité jointe sur la gamme. Cette démarche - construire une telle distribution à partir de taux - est appelée probabilisation de la gamme.

Cette distribution n'est pas connue, mais elle est spécifiée partiellement par la donnée de certaines de ses projections sur des variables (on parle de distributions marginales) et par la définition de la gamme. Il est possible qu'elle soit sur-spécifiée, c'est-à-dire qu'il n'existe aucune distribution de probabilité sur la gamme dont les marginales correspondent aux taux renseignés, ceci à cause des contraintes définissant la gamme. Dans le cas contraire, elle est généralement sous-spécifiée : plusieurs distributions jointes différentes peuvent produire les mêmes marginales. Dans ce cas la probabilité d'un véhicule n'est pas connue avec précision, mais l'on peut en connaître la plus petite et la plus grande valeurs possibles. De la même façon, plusieurs distributions peuvent être envisagées pour les marginales (pour les taux) non renseignées. Les requêtes en rapport avec la probabilisation de la gamme sont :

Cohérence des taux : Les taux sont-ils cohérents (existe-t-il une distribution de probabilité sur la gamme dont les marginales sont précisément les taux renseignés) ?

Taux conflictuels : Si les taux sont incohérents, en exhiber un sous-ensemble (minimal) incohérent modulo la définition de la gamme.

Configuration des taux : Si les taux sont cohérents, calculer une borne min et une borne max de la probabilité de l'affectation d'une valeur à une variable de véhicule. Utilisé itérativement avec la définition de taux, ce mécanisme permet de configurer progressivement un jeu de taux qui reste cohérent à toutes les étapes de la construction.

Extrapolation de taux : Si les taux sont cohérents, calculer une borne min et une borne max de la probabilité d'une expression booléenne sur des affectations - typiquement, sur des cas d'emploi. Ce calcul permet notamment d'encadrer les volumes prévisionnels de pièces.

Génération d'échantillon : Les taux étant cohérents, générer un échantillon de n véhicules conformément à la distribution de probabilité sous-jacente ; si elle est sous-spécifiée, on peut se baser sur la distribution d'entropie maximale. Cet échantillon sera utilisé comme une commande prévisionnelle par d'autres logiciels (simulation de la production, calcul de prix de revient, etc.).

3.6 Temps de réponse

Pour les applications interactives (typiquement, la configuration en ligne) le temps de réponse doit être (au pire) de l'ordre de la seconde. Les applications hors ligne, elles, peuvent compter leur temps de calcul en heures. Mais si elles ont un grand nombre de requêtes à effectuer, cela signifie que chaque requête devra également prendre un temps réduit (entre la milliseconde et la seconde, suivant les cas). Cependant, il reste possible de consacrer un temps beaucoup plus élevé (par exemple se comptant en minutes) pour les requêtes les plus difficiles, tant que cela ne modifie pas l'ordre de grandeur du temps de calcul total. C'est cette logique qui prévaut pour les contrôles de cohérence de la documentation véhicule et de la documentation pièces.

4 Formalisation des requêtes ; complexité

Nous étudions ici comment les requêtes présentées en Section 3 peuvent être formalisées en termes de CSP, et présentons nos premiers résultats quant à leur complexité théorique.

4.1 Notations et définitions

CSP Un *CSP* est, on l'a vu, un triplet $P =< X, D, C >$. Pour toute variable x de X, $D(x)$ est le domaine (supposé fini) de x. Pour toute contrainte c de C, on note $vars(c)$ l'ensemble des variables dont elle restreint les valeurs.

Une *affectation* d est une fonction qui associe à chaque $x \in X$ une valeur de son domaine ou le sym-

bole $*$, qui indique que la variable n'a pas été affectée. d est complète (resp. partielle) si le symbole $*$ n'est pas (resp. est) présent. Plus largement, on dit que d affecte $Y \subseteq X$ (resp. c) ssi pour tout $x \in Y$ (resp. pour tout $x \in vars(c)$), $d(x) \neq *$. $D(Y) = \{d, \forall x, d(x) = * \Leftrightarrow x \notin Y\}$ est l'ensemble des affectations de $Y \subseteq X$. Enfin, on note $vars(d)$ les variables affectées dans d, et on dit que d' étend d (à $vars(d')$) ssi $var(d) \subseteq vars(d')$ et $\forall x \in vars(d), d(x) = d'(x)$; ceci est noté $d' \models d$.

Une contrainte peut être vue comme une fonction c de l'ensemble des affectations de $vars(c)$ dans $\{\top, \bot\}$: $c(d) = \top$ ssi d satisfait la contrainte. On ne fait pas d'hypothèse sur la manière dont sont représentées les contraintes, mais on suppose que, pour toute affectation d affectant (au moins) toutes les variables de c, on peut tester en temps polynômial (idéalement, linéaire) si d satisfait c (noté $d \models c$) ou la viole (noté $d \not\models c$). Une *solution* d'un CSP est une affectation complète de ses variables qui satisfait toutes ses contraintes. On note $Sols(P)$ l'ensemble des solutions de P. Si $Sols(P) = \emptyset$, P est dit *incohérent*, sinon, il est dit *cohérent*.

$Sols(P)^{\downarrow Y} = \{d \in D(Y)$ t.q. $\exists d' \in Sols(P), d' \models d\}$ dénote la *projection* de $Sols(P)$ sur $Y \subseteq X$.

On dit que v est une valeur *globalement cohérente* pour x ssi $v \in Sols(P)^{\downarrow\{x\}}$. Lorsque toute les valeurs des domaines du CSP sont globalement cohérentes pour la variable correspondante, on dit que les domaines du CSP sont globalement cohérents (ou simplement que le CSP est globalement cohérent).

Conditions Booléennes Un *fluent* est un couple (x, A), avec $x \in X$ (et généralement, $A \subseteq D(x)$, mais pas nécessairement) ; il est élémentaire quand A est un singleton - il représente alors une affectation de x. Une *condition booléenne* est une formule logique (utilisant les opérateurs $\vee, \wedge, \Longrightarrow, \neg, \Leftrightarrow$ dans leur acception classique) dont les atomes sont des fluents. $vars(c)$ dénote l'ensemble des variables de c.

Une affectation d telle que $x \in vars(d)$ satisfait le fluent $f = (x, A)$ si et seulement si $d(x) \in A$ (on note $f(d) = \top$). Soit c une condition booléenne et d une affectation de $vars(c)$. La valeur de vérité de c dans d, notée $c(d)$ est calculée conformément à l'interprétation habituelle des opérateurs logiques. d satisfait c (noté $d \models c$) ssi $c(d) = \top$.

Certaines contraintes de la gamme seront représentées par des conditions booléennes - typiquement les contraintes options (c.f. section 2). Les description synthétiques utiles aux tâches de restitution de la gamme (c.f. Section 3.2) et les cas d'emplois relatifs à la documentation pièces (c.f. Section 3.3)le seront également. Dans la suite, on suppose généralement que les conditions booléennes manipulées sont intrinsèquement cohérentes ($\exists d \in D(vars(c)), c(d) = \top$) et que l'on peut tester en temps polynômial (idéalement, linéaire) leur satisfaction par une affectation de leurs variables.

4.2 Résultats

Les Tables 1 à 4 décrivent comment les requêtes présentées en section 3 peuvent être formalisées, et donnent nos premiers résultats quant à leur complexité. Certaines définissent clairement des problèmes de décision au sens de la théorie de la complexité, par exemple le problème de cohérence de la gamme (il revient à déterminer si $< X, D, C >$ est cohérent). D'autres sont des problèmes de recherche souvent NP-difficiles. Dans les Tables qui suivent, nous essayons d'affiner cette information : la complexité d'un problème de recherche d'objet (ex : d'une solution d'un CSP, d'un ensemble minimal incohérent de contraintes) est en effet directement liée à celle de l'existence d'un tel objet ou à celle du test de conformité de l'objet. Il peut arriver que le test de conformité soit linéaire alors que le test d'existence est difficile (typiquement, pour l'existence / le test de solution d'un CSP) ; il peut également arriver que le test d'existence soit trivial, alors que le test de conformité est difficile (par exemple, pour l'existence / le test d'ensembles (minimaux) incohérents de contraintes). Faute de place, seules les preuves de complexité les moins évidentes seront évoquées[2].

Documentation véhicule (voir Table 1). La requête de *Cohérence* (éventuellement contextuelle) correspond au test de cohérence, classique, d'un CSP, d'où sa NP-complétude. Sans surprise, on appellera *conflit* un ensemble incohérent de contraintes, et l'objectif est de connaître des conflits minimaux pour l'inclusion. On retrouve des notions de sous ensemble minimaux incohérents étudiées depuis la fin des années 80 en IA et plus récement en configuration [3][4][1] [6] (voir en particulier [4][6] pour le cas contextuel). La recherche de conflits minimaux (éventuellement contextuels) est NP-difficile, puisque le simple test d'(in)cohérence de $< X, D, C' >$ appelle la résolution d'un problème de cohérence. D'après [1], la question est BH_2-complète.[3]

La question de la cohérence positive du modèle est évidement une question de cohérence globale des contraintes et des domaines. Elle est donc NP-difficile.

[2] Pour plus de détails voir `ftp://ftp.irit.fr/IRIT/RPDMP/PapersFargier/wsecai10.pdf`

[3] BH_2 est l'ensemble des problèmes qui se ramènent à un problème de NP et un problème de CO-NP. Le problème BH_2-complet canonique est le problème SAT-UNSAT : il s'agit de déterminer si, étant donnée une paire (ϕ, ψ) de 3-CNF, la première est satisfiable et la seconde insatisfiable).

Nom	Donnée	Requête	Complexité
Cohérence de la gamme	$< X, D, C >$	$< X, D, C >$ admet-il une solution ?	NP-complet
Cohérence contextuelle	$< X, D, C >$ c une cond. booléenne cohérente	$< X, D, C \cup \{c\} >$ admet-il une solution ?	NP-complet
Conflits	$< X, D, C >$ $C' \subseteq C$	Déterminer si (i) $< X, D, C' >$ incohérent et (ii) $\forall C" \subsetneq C'$, $< X, D, C" >$ cohérent	BH_2-complet
Conflits contextuels	$< X, D, C >$ Cxt un ensemble de cond. booléennes cohérentes , $C' \subseteq C$	Déterminer si (iii) $< X, D, C' \cup Cxt >$ incohérent et (iv) $\forall C" \subsetneq C'$, $< X, D, C" \cup Cxt >$ cohérent	BH_2-complet
problème de recherche :	*trouver un /tous les C'*	*respectant (i) et (ii) (resp. (iii) et (iv))*	
Cohérence positive	$< X, D, C >$ Cohérent $c \in C$ une contrainte d'arité inférieure à k	Déterminer si c est globalement cohérente $(\forall d \in D(vars(c)), d \models c$ $\Rightarrow \exists d' \in Sols(P), d' \models d)$	NP-complet y compris si c est unaire

Tab. 1 – Résultats : Élaboration de la documentation véhicule

Nom	Donnée	Requête	Complexité
Validité de description	$P =< X, D, C >$ cohérent, c une cond. booléenne cohérente	Déterminer si c est valide (i.e. si $\forall d \in Sols(< X, D, C >), d \models c)$	CO-NP complet
Simplification de description	c une cond. booléenne cohérente	Déterminer si c est minimale $\forall x \in vars(c), \exists d, d' \in D(vars(c))$ t.q. $(\forall y = xd(y) \neq d'(y)) \wedge (d \models c)(\wedge d' \not\models c))$	$O(d^{\lvert vars(c) \rvert})$
Génération de description	$P =< X, D, C >$ cohérent, c une cond. booléenne cohérente	Déterminer si c équivaut à $Sols(P)^{\downarrow vars(c)}$ (si $\forall d \in D(vars(c))$: $c(d) = \top \Leftrightarrow d \in Sols(P)^{\downarrow vars(c)})$	Π_2^P-complet
problème de recherche :	*trouver un c t.q.*	$vars(c) \subseteq Y$ *et c équivaut à $Sols(P)^{\downarrow vars(c)}$*	
Comptage	$P =< X, D, C >$, cohérent, $Y \subseteq X$	Calculer $\lvert Sols(P)^{\downarrow Y} \rvert$	#P-difficile

Tab. 2 – Résultats : Restitution de la gamme

Nom	Donnée	Requête	Complexité
Concision de la doc.	$P =< X, D, C >$, cohérent C' un ensemble de cond. booléennes cohérentes	Déterminer si C' est concis $(\forall c' \in C', < X, D, C \cup \{c'\} >$ cohérent)	NP-complet
Exhaustivité de la doc.	$P =< X, D, C >$, cohérent C' un ensemble de cond. booléennes cohérentes	Déterminer si C' est exhaustif (si $< X, D, C \cup \bigcup_{c' \in C'} \{\neg c'\} >$ est incohérent)	CO-NP complet
problème de recherche	*trouver une cond. booléenne cohérente c*	*telle que* $d \models c \Leftrightarrow (\forall c' \in C', d \not\models c')$	NP-difficile
Non-dualité des emplois	$P =< X, D, C >$, cohérent C' un ensemble de cond. booléennes cohérentes	Déterminer si C' comporte des cas de dualité (si $\exists c' \neq c" \in C'$ t.q. $< X, D, C \cup \{c', c"\} >$ cohérent)	NP-complet
problème de recherche	*trouver une cond. booléenne cohérente c*	*telle que étant donnés* $c', c'' \in C'$: $\forall d \in Sols(P) : (d \models c \Leftrightarrow d \models c'$ et $d \models c")$	NP-difficile

Tab. 3 – Résultats : élaboration de la documentation pièces

Nom	Donnée	Requête	Complexité
Cohérence globale	$P =< X, D, C >$, cohérent $x \in X \setminus vars(d), v \in D(x)$	Existe t il $d' \in Sols(P)$ t.q. $d'(x) = v$?	NP-complet
Cohérence globale (version incrémentale)	$P =< X, D, C >$, glb. cohérent d tq. $\exists d' \in Sols(P), d' \models d$ $x \in X$, $v \in D(x)$	Existe t il $d' \in Sols(P)$ $d' \models d$ et $d'(x) = v$?	NP-complet
Estimation du prix	$P =< X, D, C >$, cohérent S un ens. fini de fct de coût sur X d une affectation (partielle)	Calculer $\alpha_m = \min_{d' \in Sols(P), d' \models d} \Sigma_{s \in S} s(d')$ $\alpha_m = \max_{d' \in Sols(P), d' \models d} \Sigma_{s \in S} s(d')$	NP-difficile NP-difficile
Estimation du délai	$P =< X, D, C >$, cohérent x_t une variable codant le délai d une affectation (partielle)	Calculer $\alpha_m = \min_{d' \in Sols(P), d' \models d} d'(x_t)$ $\alpha_m = \max_{d' \in Sols(P), d' \models d} d'(x_t)$	NP-difficile NP-difficile
Calcul de restauration	$P =< X, D, C >$ $Choix$ un ens. de cond. booléennes $R \subseteq Choix$	Est il vrai que $< X, D, C \cup R >$ est cohérent et que $\forall R' \subsetneq Choix$ t.q. $R \subsetneq R', < X, D, C \cup R' >$ est incohérent ?	BH-2 complet
Calcul de conflit	$P =< X, D, C >$ $Choix$ un ens. de cond. booléennes $R \subseteq Choix$	Est il vrai que $< X, D, C \cup R >$ est incohérent et que $\forall R' \subsetneq R$ $< X, D, C \cup R' >$ est cohérent ?	BH-2 complet
Complétion	d une affectation (partielle) $P =< X, D, C >$ glb. coherent pour d	Existe-t-il $d' \in Sols(P)$ t.q. $d' \models d$?	trivial
Problème de recherche		*Trouver* $d' \in Sols(P)$ *t.q* $d' \models d$	? ? ? ? ?
Problème de recherche *version optimisation*	*S un ens. fini de fct de coût*	*Trouver* $d' \in Sols(P)$ *tel que* $d' \models d$ *et que* $\forall d" \in Sols(P)$ si $d" \models d$ alors $\Sigma_{s \in S} s(d') \leq \Sigma_{s \in S} s(d")$	NP-Difficile
	x_t *une variable codant le délai*	*Trouver* $d' \in Sols(P)$ *t.q.* $d' \models d$ *et* $\forall d" \in Sols(P), d" \models d \Rightarrow d"(x_t) \leq d'(x_t)$	NP-difficile

TAB. 4 – Résultats : Configuration en ligne

Restitution de la gamme (voir Table 2). En restitution de la gamme, on cherche à obtenir des descriptions synthétiques sur des sous-ensembles de variables - techniquement, ce sont des projections de l'ensemble des solutions du CSP sur ces sous-ensembles. Idéalement, on les voudrait les plus simples possibles, c'est-à-dire impliquant le moins de variables possible. Dans le contexte de notre application, ces description doivent être données sous la forme d'une condition booléenne cohérente (la cohérence interne des descriptions ne doit pas être une source de complexité). Techniquement, nous dirons donc qu'une description booléenne c est *valide* ssi $\forall d \in Sols(P), d \models c$, qu'elle est *synthétique* si elle coïncide avec la projection de la gamme sur ses variables ($\forall d \in D(vars(d)), c(d) = \top \Leftrightarrow d \in Sols(P)^{\downarrow vars(c)}$) et qu'elle est minimale si il n'existe pas de $Y \subsetneq vars(C)$ telle que la restriction de c à Y lui soit équivalente. Le problèmes de *validité* est CO-NP complet, les problème d'inférence clausale ou de redondance de contrainte en étant des cas particuliers.

Le test de conformité associé à la *Génération de description* est Π_2^P-complet[4]. Le principe de la démonstration de est dans notre cas : c est la condition $\bigwedge_{x \in Y}(x, D(x))$: cette condition sélectionne un certain nombre de variables (celles de Y) sans restreindre les valeurs. Dans ce cas c équivaut à $Sols(< X, D, C >)^{\downarrow vars(c)}$ si et seulement si, pour toute affectation de Y, il existe une affectation de $X \setminus Y$ qui satisfasse le CSP. On peut donc réduire le problème canonique de Π_2^P ($\forall Y_1, \exists Y_2 : \Sigma$, Σ étant une CNF) au problème de test de l'équivalence entre la projection sur Y_2 d'un CSP représentant la CNF et la condition $c = \bigwedge_{x \in Y_2}(x, \{true, false\})$. Le test d'appartenance à Π_2^P est plus simple. L'implication $d \in Sols(< X, D, C >)^{\downarrow vars(c)} \implies c(d) = \top$ se falsifie avec un oracle (deviner d, vérifier que d satisfait toutes les contraintes puis vérifier que $c(d) = \bot$). Pour falsifier $c(d) = \top \implies d \in Sols(< X, D, C >)^{\downarrow vars(c)}$, il faut deviner une affectation de $vars(c)$, montrer qu'elle satisfait c, puis montrer qu'elle est inconsistante avec le CSP (ce qui est un problème de CO-NP).

Les problèmes de décision liés à la restitution de la gamme sont typiquement des problèmes d'inférence. Cela dit, la projection et la simplification sont basiquement des problèmes de recherche auxquels on peut ré-

[4]Rappelons qu'un problème appartient à $\Pi_2^p = \mathsf{NP}^{\mathsf{CO-NP}}$ si on peut résoudre son co-problème en (1) faisant appel à un NP-oracle et (2) établissant un certificat non pas de manière polynomiale mais en résolvant un problème de CO-NP. Le problème canonique de Π_2^p est celui qui consiste à déterminer si une formule booléenne quantifiée de la forme $\forall Y_1\ \exists Y_2\ \phi$ est valide , ϕ étant une 3-CNF et $\{Y_1, Y_2\}$ une partition de ses variables.

pondre par des algorithmes d'élimination de variables - en utilisant un espace potentiellement exponentiel.

Élaboration de la documentation pièces (voir Table 3). L'idée est de représenter chaque cas d'emploi par une condition booléenne cohérente (la cohérence interne d'un cas d'emploi n'est pas une source de complexité), et de considérer l'ensemble C' des cas d'emploi d'une pièce générique : C' est exhaustif ssi $\forall d \in Sols(P), \exists c' \in C'$ t.q. $d \models c'$; on dit que $c', c'' \in C'$ forme une paire d'emplois duale ssi $\exists d \in Sols(P), (d \models c')$ et $(d \models c'')$. C' est non exhaustif ssi $< X, D, C \cup \bigcup_{c' \in C'} \{\neg c'\} >$ est cohérent et il est non dual ssi chacun des problème $< X, D, C \cup \{c', c''\} >$ est incohérent (où $c', c'' \in C'$) . Enfin, la description de la pièce générique est concise ssi chacun des problèmes $< X, D, C \cup \{c'\} >$, $c' \in C'$ est cohérent.
Ces trois questions demandent de tester la cohérence ou l'incohérence d'un CSP, cohérent au départ, auquel on ajoute une ou plusieurs contraintes. Les preuves de complexité associées ne présentent aucune difficulté.

Enfin, on veut caractériser par une condition booléenne c les véhicules de la gamme "en erreur" par rapport à l'exigence de non-dualité ou d'exhaustivité de la pièce générique. c représente les cas de non-exhaustivité ssi $\forall d \in Sols(P), d \models c \Leftrightarrow \forall c' \in C, d$ viole c' ; il représente les cas de non dualité ssi $\forall d \in Sols(P), d \models c \Leftrightarrow \exists c', c'' \in C, d \models c'$ et $d \models c''$. Dans le premier cas, la CO-NP difficulté se prouve par réduction du problème d'inférence clausale ($C = \emptyset$, c est la clause à inférer et les c' sont les négations des clauses de la CNF). La CO-NP difficulté du second problème se prouve par réduction du même problème (c est la clause à inférer et c' et c" représentent une partition des clauses).

Configuration en ligne (voir Table 4) La *cohérence globale* des domaines doit être assurée en amont de la configuration utilisateur, pour ne lui laisser comme choix de configuration que des valeurs d'option présentes dans au moins un véhicule de la gamme. La version incrémentale est la répétition de ce principe en cours de configuration : l'utilisateur fixe une affectation d, cohérente, et l'on veut, pour toute variable x, ne laisser que des valeurs v pouvant effectivement conduire à un véhicule d' tel que $d' \models d$ et $d'(x) = v$. Dans ses deux versions, le problème de la maintenance de la cohérence globale est un problème NP-difficile.

Pour la prise en compte des coûts, on utisera une modélisation proche de celle des problèmes à contraintes souples (c.f. [12, 2]) : une "fonction de coût" est une fonction s sur $D(vars(s))$ prenant ses valeurs dans une échelle, ici $\mathbb{Z}$ et le "coût" d'une affectation est la somme des coûts qui lui sont associés par ces fonctions. Notons que certains coûts pouvant être négatifs, il faut théoriquement relaxer l'exigence de monotonie de l'opérateur d'agrégation (ici, l'addition). Lorsque tous les coûts sont positifs, que d n'instancie aucune variable et que $C = \emptyset$, le problème *d'estimation du prix* est un problème classique de minimisation de coûts dans un CSP souple, réputé NP-difficile.

Pour *l'estimation du délai*, on peut représenter le délai par une variable particulière x_t et coder les connaissances par des conditions logiques du type $c \implies x_t \in I_1 \cup \cdots \cup I_k$, les I_j étant des intervalles (ex : $x_t \in (-\infty, 8] \cup [10, +\infty)$). L'estimation du délai min (et du délai max par ailleurs) est un problème NP-difficile : lorsque d n'instancie aucune variable, et que seules des bornes inférieures sont données quant au délai, on reconnaît le problème d'optimisation dans les CSP à priorité [11] où il faut minimiser la priorité de la plus importante des contraintes violées.

La complétion de configuration est au sens d'un critère de coût ou de délai est évidement un problème NP-difficile. En revanche, on ne sait pas si il existe un algorithme polynomial permettant d'obtenir une solution à partir d'un problème globalement cohérent.Si un tel algorithme existe (ce qui est peu probable), il ne procède pas par réétablissement de la cohérence globale. Notre résultat sur la cohérence globale incrémentale montre en effect que, sous l'hypothèse P $\neq$ NP, il n'existe pas d'algorithme polynomial ré-établissant la cohérence globale suite à l'affectation d'une variable.

Pour les problèmes de *calcul de restauration et de conflits*, voir [1].

4.3 Probabilisation d'un CSP

Les problèmes de gestion de taux ne peuvent pas être modélisés de manière simple sous la forme d'un CSP ou d'un CSP valué, les taux n'étant pas indépendants les uns des autres. Il faudrait pouvoir définir une notion de CSP probabilisé comme étant un quadruplet $P^* =< X, D, C, F >$, où $< X, D, C >$ est un CSP et F est un ensemble de distributions de probabilités sur des sous-ensembles de X. Pour tout $p \in F$, on note $vars(p) \subseteq X$ les variables sur lesquelles porte p.

On dira que P^* est *cohérent* avec F si et seulement si il existe une distribution de probabilité p^* sur les affectations complètes de X telle que :

- $d \notin Sols(< X, D, C >) \implies p^*(d) = 0$
- $\forall p \in F$, d tel que $vars(d) = vars(p)$:
  $p(d) = \Sigma_{d' \in Sols(<X,D,C>), d' \models d}\ p^*(d')$

On note $Sols(P^*)$ l'ensemble des distributions de probabilité p^* vérifiant les deux conditions ci dessus. Le CSP probabilisé peut être sur-spécifié (aucune distribution n'est cohérente avec les taux de F) ou sous-spécifié, dans ce cas $Sols(P^*)$ contient plusieurs p^* .

La requête principale n'est pas le calcul explicite d'un p^*, qui est une distribution sur un ensemble de taille exponentielle ($Sols(< X, D, C >)$). Dans notre application, on cherche plutôt à connaître la probabilité d'affectations de X, de valeurs v dans les domaines, ou plus généralement, d'événements. Pour toute condition booléenne c, on peut définir :
$P_{inf}(c) = Inf_{p^* \in Sols(P*)} \Sigma_{d' \in Sols(<X,D,C>), d' \models c}\ p^*(d')$
$P_{sup}(c) = Sup_{p^* \in Sols(P*)} \Sigma_{d' \in Sols(<X,D,C>), d' \models c}\ p^*(d')$

Notons que P_{inf} et P_{sup} ne sont pas des mesures de probabilité. Elle encadrent la mesure de probabilité.

La notion de CSP probabilisé est bien différente des notions de CSP stochastique [14] ou probabiliste[5]. Dans ces deux formalismes (1) il existe deux types de variables, les variables de décision et les variables aléatoires (dans un CSP probabilisé, toutes les variables sont aléatoires) (2) les variables aléatoires sont indépendantes (il existe une unique p^* et son calcul est immédiat) et (3) on ne calcule pas la probabilité d'une affectation mais la probabilité qu'une affectation satisfasse le CSP : plusieurs affectations peuvent avoir une probabilité de 1 (dans un cSP probabilisé la somme des probabilités des affectations est égale à 1).

Les CSP probabilisés seraient une généralisation des réseaux bayésiens, relachant l'hypothèse qu'il existe une *unique* distribution jointe dont les projections seraient les distribution $p \in F$ (les Tables du réseau bayésien), et que le réseau est acyclique. Par conséquent le simple calcul d'une affectation de probabilité inférieure maximale serait un problème NP-difficile.

Sans entrer plus dans les détails, disons simplement que l'analyse de la tâche de gestion des taux en termes de programmation par contraintes fait apparaître le besoin d'un nouveau modèle qui serait une généralisation de réseaux bayesiens. L'analyse de la complexité des requêtes associées et la définition d'algorithmes efficaces sortent du cadre de cet article.

5 Conclusion

La formalisation d'un certain nombre d'applications métier utilisant une modélisation à base de contraintes d'une gamme montre que la preuve de la cohérence d'un CSP est loin d'être la seule requête que doivent addresser les algorithmes de programmation par contraintes : on rencontre notamment des problèmes d'inférence, des problèmes d'optimisation, des problèmes de comptage et des problèmes de marginalisation, qui se situent souvent au dessus de NP.

Dans la plupart des cas, le CSP considéré est notoirement cohérent, connu à l'avance, et commun à toute une série de requêtes. Chez Renault, l'équipe "Intelligence Artificielle Appliquée " en charge de ces applications a choisi une approche par compilation du CSP représentant la gamme [9], approche qui permet de répondre en temps satisfaisant (de l'ordre du centième de seconde pour la configuration en ligne, par exemple) à la plupart des requêtes.

La question est maintenant de savoir (1) si les algorithmes proposés par la communauté CSP peuvent donner des résultats comparables, (2) si la communauté peut proposer des algorithmes efficaces pour les problèmes qui ne sont pas typiquement des problèmes de satisfaction (par exemple, l'établissement de la cohérence globale, le calcul de marginalisations), et (3) comment étendre le modèle CSP pour qu'il sache traiter de questions d'inférence complexes - typiquement, effectuer de l'inférence probabiliste.

Références

[1] J. Amilhastre, H. Fargier, and P. Marquis. Consistency restoration and explanations in dynamic CSP - application to configuration. *Artificial Intelligence*, 135(1-2) :199–234, 2002.

[2] M. C. Cooper and T. Schiex. Arc consistency for soft constraints. *Artificial Intelligence*, 154(1-2) :199–227, 2004.

[3] Johan de Kleer. Problem solving with the atms. *Artificial Intelligence*, 28(2) :197–224, 1986.

[4] J. L. de Siqueira N. and J-F Puget. Explanation-based generalisation of failures. In *ECAI*, pages 339–344, 1988.

[5] H. Fargier and J. Lang. Uncertainty in constraint satisfaction problems : a probalistic approach. In *ECSQARU*, pages 97–104, 1993.

[6] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Stumptner. Consistency-based diagnosis of configuration knowledge bases. *Artif. Intell.*, 152(2) :213–234, 2004.

[7] E. Gelle and R. Weigel. Interactive configuration using constraint satisfaction techniques. In *PACT-96*, pages 37–44, 1996.

[8] S. Mittal and B. Falkenhainer. Dynamic constraint satisfaction problems. In *AAAI*, pages 25–32, 1990.

[9] B. Pargamin. Vehicle sales configuration : the cluster tree approach. In *ECAI'02 Configuration Workshop*, 2002.

[10] D. Sabin and E. C. Freuder. Configuration as composite constraint satisfaction. In *AI and Manufacturing Research Planning Workshop*, pages 153–161, 1996.

[11] T. Schiex. Possibilistic constraint satisfaction problems or "how to handle soft constraints ?". In *UAI*, pages 268–275, 1992.

[12] T. Schiex. Arc consistency for soft constraints. In *CP*, pages 411–424, 2000.

[13] T. Walsh. Sat v csp. In *CP*, pages 441–456, 2000.

[14] T. Walsh. Stochastic constraint programming. In *ECAI*, pages 111–115, 2002.