CENTRE DE RECHERCHE
EN INFORMATIQUE DE LENS

De la résolution du problème SAT à la résolution de problèmes autour de SAT

Mémoire d'

Habilitation à Diriger des Recherches

Université d'Artois

(Spécialité Informatique)

présentée et soutenue publiquement le 15 décembre 2000

par

Lakhdar SAÏS

Composition du jury

Rapporteurs :	Michel CAYROL	Professeur, Université Paul Sabatier, Toulouse
	Jean-Jacques CHABRIER	Professeur, Université de Bourgogne, Dijon
	Marie-Catherine VILAREM	Professeur, Université de Montpellier II
Examinateurs :	Éric GRÉGOIRE	Professeur, Université d'Artois (directeur de recherche)
	Gérard FERRAND	Professeur, Université d'Orléans
	Pierre MARQUIS	Professeur, Université d'Artois
	Pierre SIEGEL	Professeur, Université de Provence, Aix-Marseille I

UNIVERSITÉ D'ARTOIS – CENTRE DE RECHERCHE EN INFORMATIQUE DE LENS
rue de l'Université, SP 16 – F-62307 LENS CEDEX
Tél : +33 (0)3 21 79 32 74 – Fax : +33 (0)3 21 79 32 72

Remerciements

Merci à Michel CAYROL, Jean-Jacques CHABRIER et Marie-Catherine VILAREM d'avoir accepté d'examiner ce modeste travail, malgré vos nombreuses activités et responsabilités. Par votre présence dans les projets « *Bahia* » et « *Classes polynômiales* » de l'ex PRC-IA, j'ai énormément appris à la fois sur le plan scientifique, mais aussi humain.

Merci, à Gérard FERRAND, professeur d'informatique à l'Université d'Orléans de m'avoir fait l'honneur de participer à ce jury.

Merci à Éric GRÉGOIRE de m'avoir fait une grande place au sein de son laboratoire, d'avoir partagé avec moi l'aventure SAT, celle autour de SAT et pour bien d'autres choses. Je lui dois beaucoup.

Merci à Pierre MARQUIS pour m'avoir initié à la compilation des bases de connaissances, et surtout pour avoir enrichi après chaque discussion ma propre base de connaissances. Je lui suis reconnaissant.

Merci à Pierre SIEGEL de m'avoir permis de faire ma thèse d'Université dans sa jeune, dynamique et chaleureuse équipe « raisonnement et preuve » de l'Université de Provence, et de m'honorer de sa présence dans ce jury.

J'ai beaucoup aimé la « *pastaga* », en particulier la « *moresque* », surtout quand on la boit en bonne compagnie et en face de la « Bonne Mère ». À tous les anciens de l'UFR-MIM de l'Université de Provence, je vous dis tout simplement merci pour votre gentillesse, votre solidarité, et pour tous les moments inoubliables que nous avons passés ensemble.

Chèr(e)s ami(e)s marseillais(es), maintenant je supporte le Racing Club de Lens. Qu'il pleuve ou qu'il neige, que Lens gagne ou perde, il y a toujours au moins 30 mille spectateurs au stade Bollaert. Leur attachement indéfectible au RCL montre mieux que tout autre exemple les qualités des gens du Nord.

À Lens, j'ai découvert les vertus d'un bon digestif. Pour ne pas être pris pour ce que je ne suis pas, j'ajoute la mention « *à boire avec modération* ».

À tous les collègues du CRIL, et à tous les membres de l'IUT de Lens, de l'accueil, jusqu'à mon bureau, en faisant le tour par le quatrième département, merci. J'ajouterai une mention spéciale à Assef, Bertrand, Jean-Luc et Sylvie pour m'avoir soulagé ces derniers temps de mes tâches quotidiennes et d'autres plus conjoncturelles.

Bertrand, je tiens à t'exprimer ici ma profonde amitié. Comme je ne pourrai jamais te remercier assez, je te dis simplement merci et « *plus vite que ça !* ».

À Souhila, Ouamar et Massin, je vous demande pardon pour mes longues absences, et je vous promets que cela changera et, surtout, qu'après les fêtes « *adieu la cigarette* ».

Je termine par ceux qui m'ont soutenu malgré la distance, mes parents, mes sœurs et mon frère. Sachez que par mes pensées et mon cœur je suis souvent avec vous. Merci de m'avoir appris le métier de berger avant de rejoindre les bancs de l'école.

À mes grands-parents

Table des matières

Avant-propos

Avant de commencer la rédaction de ce document, j'ai naturellement feuilleté de nombreux mémoires d'habilitation à diriger des recherches. Je ne déroge pas à la tradition de situer dans le temps les environnements dans lesquels j'ai évolué, en profitant pour remercier les personnes que j'ai eu la chance de rencontrer. Je me limiterai dans cet-avant propos aux personnes ayant influencé directement mon itinéraire scientifique.

– je n'aurais pas fait d'études sans mon grand-père OUAMAR grâce à qui *l'école primaire* du village a vu le jour. Ce mémoire est pour toi !

– mes premiers pas en recherche ont été effectués en fin de cycle d'ingénieur à *l'Institut National d'Enseignement Supérieur en Informatique de l'Université de Tizi-Ouzou*. L'étude que j'ai réalisée sous la direction du Professeur Md Said AIDENE, concerne la *résolution des problèmes de programmation linéaire à grand nombre de paramètres (de grande taille)*, en utilisant différentes techniques de décomposition. Ce projet m'a permis de découvrir un domaine très riche "la recherche opérationnelle" qui a des liens étroits avec mes centres d'intérêts actuels. Merci MOHAND pour la documentation en russe que vous avez eu la gentillesse de me traduire.

– Merci au Professeur Henri MOREL (un grand Monsieur) pour avoir retenu ma candidature au DEA de Mathématiques et d'Intelligence Artificielle de *l'Université d'Aix-Marseille II, Luminy* et de m'avoir initié au problème SAT (Satisfiabilité d'une formule booléenne mise sous forme normale conjonctive), aux techniques de résolution (SL-Résolution, la procédure de Davis et Putnam), etc.

– Une belle aventure avec mon directeur de thèse Pierre SIEGEL a commencé en 1990 au *Laboratoire d'Informatique d'Université de Provence (LIUP)*, par une petite histoire de pigeons (tiroirs et chaussettes). C'était l'époque des puzzles logiques, des reines, des problèmes de Ramsey, pigeons, etc. Ces problèmes présentent pour une large part des structures très symétriques. La prise en compte de ces informations structurelles lors de la résolution a permis un gain considérable en efficacité. Conscient du lien entre codage et résolution de problèmes, nous avons proposé un formalisme (à cardinalité) permettant une expression plus naturelle et compacte des problèmes et de leurs propriétés de symétries. Cette étude a été conduite en collaboration étroite et quotidienne avec Belaid BENHAMOU que je remercie chaleureusement.
  Je raconte une partie de cette belle aventure qui est loin d'être terminée.
  Pierre, avec le recul, je m'aperçois que tu as représenté pour moi plus qu'un directeur de thèse. Tu fus un exemple pour tes qualités humaines difficiles à énumérer. Je t'exprime ici ma profonde gratitude !
  Je ne peux conclure l'étape marseillaise sans citer Antoine RAUZY, nos discussions souvent informelles et amicales ont été très riches. C'était la période des problèmes aléatoires difficiles, des phénomènes de seuil, des algorithmes de recherche locale, etc.
– Je mesure aujourd'hui ce que des projets comme *BAHIA (Booléens Algorithmes et Heuristiques pour l'IA), classes polynomiales de l'ex PRC IA* peuvent représenter pour un jeune chercheur. J'exprime ici, ma reconnaissance à mes aînés et à tous les membres que je continue à revoir avec beaucoup de

plaisir dans le cadre des *Journées Nationales sur la résolution Pratique de problèmes NP-Complet (JNPC)* et des réunions du *groupe de travail "Axe algorithme" du PRC GdR I3*.

– J'ai rejoint *le Centre de Recherche en Informatique de Lens (CRIL)* en septembre 94. Ces six années passées à Lens ont été extrêmement riches à la fois sur le plan personnel et professionnel. Une page ne suffirait pas à exprimer ma reconnaissance et mes remerciements. J'exprime ma reconnaissance à tous les collègues qui m'ont guidé dans la visite autour de SAT:
  – logiques non monotones, révision, fusion et validation des systèmes à base de connaissances (Éric GRÉGOIRE)
  – compilation des bases de connaissances (Pierre MARQUIS et Yacine BOUFKHAD)
  – problèmes de satisfaction de contraintes (Assef CHMEISS)

  J'ajoute une mention spéciale à mes étudiants Bertrand MAZURE et Laure BRISOUX (aujourd'hui tous deux maîtres de conférences) que j'ai eu la chance de co-encadrer sous la direction d'Éric GRÉGOIRE.

En résumé, j'ai appris tout au long de ces dix dernières années que la recherche est avant tout un travail d'équipe. Je peux même dire que je n'ai rarement travaillé seul, à l'exception de l'année 1993-1994 (année séparant la fin de ma thèse de ma nomination à l'IUT de Lens). Ce qui m'amène à dire que sans toutes ces personnes, ce document n'aurait pas vu le jour. C'est dans ce sens qu'il faut comprendre le terme *nous* très employé dans ce mémoire.
Dans celui-ci, j'ai choisi volontairement de privilégier la présentation d'un domaine qui a occupé une place importante dans mes activités de recherche pour lequel ma contribution personnelle a été la plus significative : la résolution du problème SAT et plus récemment les problèmes de satisfaction de contraintes (CSP). Dans la suite, les références concernant les travaux auxquels j'ai contribués seront mentionnées en caractères **Gras**.

Organisation du mémoire

Ce document est divisé en trois parties. La *première partie* est une synthèse de mes activités scientifiques. J'ai longtemps hésité quant à la forme et à l'organisation que je devais donner à cette partie centrale du mémoire. Étant co-responsable avec Philippe JÉGOU de l'axe algorithme du GdR I3, j'ai eu récemment l'occasion de rédiger une synthèse des activités du groupe de travail ; synthèse axée sur la réponse à trois questions :

– quelles sont les avancées scientifiques récentes (résultats établis) ?
– quels sont les verrous scientifiques actuels et les sujets ou thèmes de recherche émergeants ou qui devraient être lancés ?
– quelles sont les applications majeures en regard de ces sujets ou thèmes de recherche ?

Ce travail m'a aidé au choix de l'organisation de cette partie du mémoire. En effet, en regroupant les travaux récents de la communauté autour de quatre axes : codage, résolution, extensions, et applications, je me suis aperçu que j'ai moi même travaillé sur ces différents aspects. En outre, le titre qui a été donné par Pierre MARQUIS au sous-groupe de travail du GT 1.2 dont il est le responsable s'est pour moi révélé le meilleur titre reflétant mes travaux de recherche : « Utilisation des techniques de résolution du problème SAT à la résolution de problèmes autour de SAT ». Pour exprimer ma démarche dans le temps, j'ai légèrement modifié le titre précédent en : « De la résolution du problème SAT à la résolution de problèmes autour de SAT ». Ce qui m'a conduit a structurer la première partie de ce document en quatre chapitres :

– *Introduction* : dans ce chapitre je présente le contexte de mes travaux et la démarche scientifique adoptée.
– *Problème SAT* : ici sont exposés mes travaux sur la résolution de systèmes de contraintes booléennes. Après une description du problème, un *bref* état de l'art des travaux sur SAT est donné. Je décris ensuite mes travaux autour de quatre thèmes : exploitation de propriétés structurelles, aspects liés au codage et à l'extension du formalisme propositionnel, aspects algorithmiques et enfin génération aléatoire et phénomènes de seuils. Ces différents aspects sont bien évidemment très liés. Par exemple, les propriétés structurelles ont été étudiées dans le but d'améliorer la résolution de certains problèmes structurés et difficiles. On sait aussi que le formalisme utilisé et la manière de coder un problème ont une incidence directe sur la résolution proprement dite.
– *Problèmes autour de SAT* : comme je l'ai évoqué dans l'avant-propos, et afin de rester dans le cadre de la problématique de satisfaction de contraintes, j'ai fait le choix de privilégier la présentation de mes récents travaux relatifs à la résolution des problèmes de satisfaction de contraintes (CSP). Je laisse tout de même une place aux autres travaux : compilation et révision des bases de connaissances, fusion et validation des bases de connaissances et logique du premier ordre.
– Perspectives : dans ce chapitre, je mets en évidence mes perspectives de recherche à court et à moyen termes.

L'avantage de cette organisation est de permettre une bonne structuration. L'inconvénient est de ne pas permettre de voir clairement la démarche scientifique adoptée. Je ferai en sorte de minimiser la portée de cet inconvénient.

La *seconde partie* est un *curriculum vitae* composé de quatre chapitres. Après une brève description de mon état civil, de mon cursus et de mes activités professionnelles (chapitre **??**), j'introduis au chapitre **??** mes activités de recherche (activités d'encadrement de la recherche, d'animation scientifique et liste de publications). Je présente ensuite dans le chapitre **??** mes activités administratives et autres responsabilités. Pour conclure cette partie une présentation de mes activités d'enseignement effectuées depuis septembre 1988 est donnée au chapitre **??**.

La *troisième partie* de ce mémoire regroupe un sous-ensemble de mes publications parmi les plus significatives.

Première partie

Synthèse des travaux de recherche

Chapitre 1

Introduction

Dans l'introduction de ma thèse d'Université, j'ai écrit : « *La représentation des connaissances et mécanisation des raisonnements est un thème central en Intelligence Artificielle ... Cependant dans le cadre de l'IA, on attend des résultats pratiques : la déduction doit être programmée et permettre d'obtenir des résultats dans un temps raisonnable*». Ce paragraphe très largement inspiré de celui écrit par Pierre SIEGEL dans sa thèse d'État sur la représentation et l'utilisation de la connaissance en calcul propositionnel [Siegel 1987] se retrouve d'une certaine manière dans le texte fondateur du groupe de travail GT 1.2[1] *(axe algorithmes)* du GdR I3 auquel je participe : « *il est essentiel que les formalismes utilisés pour la représentation des connaissances et la formalisation du raisonnement soient traitables pratiquement et/ou théoriquement... Cet axe couvre donc un champ qui va de l'étude théorique et pratique des problèmes NP-complets, aux procédures de preuve pour les logiques non classiques* ». L'étude théorique et pratique du problème SAT est essentielle si l'on veut disposer un jour de mécanismes de déduction efficace. Ce qui a conduit de nombreux chercheurs ayant travaillé sur les aspects modélisation du raisonnement à s'intéresser de plus près à la logique propositionnelle comme langage de représentation et au problème SAT par son lien à la déduction logique[2].

Je peux maintenir ce premier objectif aujourd'hui, d'autant que j'ai travaillé dans cette direction au cours de ces dernières années en collaboration avec Éric GRÉGOIRE et Pierre MARQUIS. Ainsi, les travaux sur la révision en logique propositionnelle non monotone, sur la compilation et la fusion de bases de connaissances intègrent parfaitement ce cadre. Mais au delà de ces objectifs, le problème SAT par sa place centrale (problème NP-complet de référence[3] [Cook 1971]) en théorie de la complexité rend son intérêt pratique encore plus important. On le retrouve en tant que sous-problème ou cas particulier dans de nombreux domaines comme la déduction automatique, les problèmes de satisfaction de contraintes, la vérification de circuit, la planification, la cryptographie, etc.

Les arguments ci-dessus montrent le caractère fondamental de ce problème. Il reste qu'on peut toujours s'interroger sur le choix d'un langage de représentation aussi rudimentaire qu'est la logique propositionnelle. Ce langage par sa simplicité permet de représenter une grande variété de problèmes et surtout d'étudier finement les algorithmes mis en œuvre. Les progrès récents obtenus autour de la résolution pratique du problème SAT (cf. chapitre 2) ont permis sans aucun doute de conforter cet argument. En effet, les succès croissants obtenus dans la résolution de problèmes réels [Reiter & Mackworth 1989, Kautz & Selman 1992, Zhang & Hsiang 1994, Crawford & Baker 1994, Larrabee 1992, Silva & Sakallah 2000, Biere *et al.* 1999, Massacci & Marraro 2000] mettent en évidence les premiers signes d'un passage à l'échelle industrielle et

1. Ce groupe de travail fait partie du thème 1 « modèles de raisonnement et algorithmes ».
2. $F \models f$ si et seulement si $(F \wedge \neg f)$ est insatisfiable.
3. SAT est le premier problème à avoir été montré complet pour la classe NP (Non déterministe Polynômial).

commerciale (« *Greentech Computing Ltd* » (UK) [Gre], « *Prover Technology AB.* » (Suède) [Prover], etc.). Pour s'en convaincre, il suffit de constater l'évolution quantitative et qualitative des bases de tests utilisées ces dernières années pour évaluer les algorithmes de résolution [Dimacs 1993, Hoos & Stützle 2000]. On peut également trouver des échos de ces résultats dans la presse « grand public » comme l'illustre un article paru dans le « New-York Times Magazine » [Johnson 1999].

Ce saut qualitatif ouvre et/ou remet au goût du jour de nombreuses perspectives (cf. chapitre 4) qui vont du codage à la résolution, en passant par l'étude des extensions possibles des cadres existants et de l'utilisation des techniques de SAT à la résolution de problèmes autour de SAT. C'est dans ce contexte que se situe une grande partie de mes travaux de recherche.

Le premier objectif de mes travaux concerne l'élargissement de la classe des instances traitables : il s'agit de pousser aussi loin que possible les limites des algorithmes existants. L'approche utilisée consiste d'une part à ajouter de nouveaux principes aux algorithmes, en exploitant par exemple, les propriétés structurelles et syntaxiques des problèmes traités, et d'autre part, à améliorer le formalisme d'expression des problèmes tout en restant dans le cadre du calcul propositionnel. Comme je l'ai déjà évoqué, il s'agit ensuite d'étendre et/ou d'utiliser certains résultats obtenus sur SAT à d'autres domaines comme la compilation et la révision des bases de connaissances ou encore les problèmes de satisfaction de contraintes.

Dans la suite, je détaille mes recherches selon deux axes :

- travaux sur le problème SAT (exploitation des propriétés structurelles, aspects codage, aspects algorithmiques, phénomène de seuil et génération aléatoire) ;
- travaux autour de SAT (problèmes de satisfaction de contraintes, compilation et représentation de l'ensemble des solutions, révision et fusion de bases de connaissances).

Chapitre 2

Problème SAT

Sommaire

2.1 Définitions de base et bref état de l'art

Une *variable booléenne* est une variable prenant ses valeurs dans l'ensemble $\{vrai, faux\}$. Soit $X = \{x_1, x_2, \ldots, x_n\}$ un ensemble de n variables booléennes. Une *affectation (partielle)* des variables de X est une fonction (partielle) $I : X \rightarrow \{vrai, faux\}$. On distingue pour chaque variable x deux littéraux, x (*un littéral positif*) et $\neg x$ (*un littéral négatif*) où $\neg$ est le symbole de négation logique. Un littéral x (resp. $\neg x$) est vrai dans I si et seulement si $I(x) = vrai$ (resp. $I(x) = faux$).

Une *formule booléenne mise sous forme normal conjonctive (CNF)* est une conjonction de m clauses [1] : $\mathcal{F} = C_1 \wedge C_2 \wedge \cdots \wedge C_m$. Chaque *clause* C_i est un ensemble de littéraux combinés par l'opérateur logique *ou* ($\vee$). Une affectation (interprétation) I est dite solution ou *modèle* de $\mathcal{F}$ si elle satisfait chaque clause de $\mathcal{F}$.

1. On désigne souvent une formule CNF comme un ensemble de clauses.

Le problème SAT est un problème de décision qui consiste à déterminer s'il existe un modèle de la formule d'entrée $\mathcal{F}$. La formule $\mathcal{F}$ est dite satisfiable si la réponse est positive et insatisfiable sinon.

On désignera par $\mathcal{F}(x)$ la formule $\mathcal{F}$ simplifiée par l'affectation de x à $vrai$. $\mathcal{F}(x) = \{C|C \in \mathcal{F}, \{x, \neg x\} \cap C = \emptyset\} \cup \{C \backslash \neg x | C \in \mathcal{F}, \neg x \in C\}$. L'ensemble de variables de $\mathcal{F}$ est noté $var(\mathcal{F})$. La taille d'une clause C_i (notée $|C_i|$) est égale au nombre de littéraux de la clause. Une clause de taille 2 (resp. 1) est dite binaire (resp. unitaire). Un littéral est dit *littéral unitaire* ou *mono-littéral* (resp. *littéral pur ou monotone*) s'il apparaît dans une clause unitaire (resp. si son opposé n'apparaît pas dans la formule)[2]. La *propagation unitaire* (resp. *des littéraux purs*) désigne le processus consistant à simplifier la formule en répétant l'opération de satisfaction des littéraux unitaires (resp. des littéraux purs). On note $k - SAT$ les instances dont les clauses sont toutes de longueur k. En fait, toute formule booléenne peut être transformée en temps linéaire en un ensemble de clauses équivalent du point de vue de la satisfiabilité. Cette transformation nécessite l'ajout de variables supplémentaires. De la même manière une formule CNF peut également se réduire à une instance de 3-SAT (une restriction de SAT qui reste NP-complète [Cook 1971]). Enfin, la taille d'une formule est donnée par $|\mathcal{F}| = \sum_{i=1}^{m} |C_i|$.

Exemple :
Donnée : Une formule booléenne mise sous forme CNF :
$$\mathcal{F} = (a \vee b \vee c) \wedge \\ (\neg a \vee b) \wedge \\ (\neg b \vee c) \wedge \\ (\neg c \vee a)$$
Question : Est-ce que la formule $\mathcal{F}$ admet au moins un modèle?
Réponse : Pour cet exemple, la réponse est oui : l'affectation $a = vrai, b = vrai, c = vrai$ satisfait la formule $\mathcal{F}$. Comme $\mathcal{F}$ admet une seule solution, on a $\mathcal{F} \wedge (\neg a \vee \neg b \vee \neg c)$ est insatisfiable.

La première méthode qui vient à l'esprit (sans être un spécialiste de SAT) pour décider si une formule $\mathcal{F}$ est satisfiable ou non est celle qui consiste à passer en revue les 2^n affectations possibles. Ce qui donne une complexité en temps dans le pire cas de l'ordre de 2^n. A l'heure actuelle, il n'existe pas de méthode permettant de décider de la satisfiabilité de toute formule de type $\mathcal{F}$ en temps polynômial en fonction de n. L'existence d'un tel algorithme est peu vraisemblable, même si sa non-existence n'a pas été prouvée. En effet, la découverte fondamentale de S.A. COOK au début des années 70 est que de nombreux problèmes (e.g. en recherche opérationnelle et en théorie des graphes) peuvent être réduis au problème SAT. Cette notion de réduction entre différents problèmes est centrale en théorie de la complexité. M.R. GAREY et D.S. JOHNSON [Garey & Johnson 1979] dans leur ouvrage présentent un large catalogue de problèmes NP-complets. La résolution efficace d'un de ces problèmes aura une incidence immédiate sur l'ensemble du catalogue.

Vu le nombre important de travaux de recherche réalisés depuis de nombreuses années sur SAT, il est très difficile, voire impossible, de réaliser un inventaire complet. Ce qui dépasserait le cadre de cette synthèse. Je me limiterai à un exposé des travaux importants et/ou en relation directe avec mes travaux de recherche. Le lecteur intéressé par un panorama et une liste bibliographique assez complète peut se reporter au récent « Survey » traitant du problème SAT [Franco *et al.* 1997].

Pour "résoudre SAT", on ne peut donc que chercher des heuristiques et de nouveaux raffinements permettant d'accélérer en pratique les traitements ou exhiber des restrictions pour lesquelles on peut garantir une résolution en temps polynômial.

2. Si x est un littéral unitaire ou pur dans $\mathcal{F}$, alors $\mathcal{F}$ et $\mathcal{F}(x)$ sont équivalents du point de vue de la satisfiabilité.

2.1.1 Les classes polynômiales du problème SAT

Une classe polynômiale est un ensemble de formules qu'il est possible de reconnaître et de résoudre en temps polynômial. Une classe polynômiale admet d'autant plus d'intérêt qu'elle recouvre une large classe de problèmes réels. Outre l'intérêt théorique de ce type d'étude, un autre bénéfice concerne l'exploitation de ces classes pour mieux traiter le cas général [Gallo & Urbani 1989] ou dans d'autres domaines comme la compilation des bases de connaissances [Selman & Kautz 1991, Marquis 1995].
Parmi les « nombreuses » classes polynômiales du problème SAT, 2-SAT [Cook 1971] (instances formées de clauses binaires) et Horn-SAT (instances formées de clauses ayant au plus un littéral positif) restent sans aucun doute parmi les plus intéressantes. En effet, les clauses binaires se retrouvent très fréquemment dans les problèmes réels, un traitement particulier sur les clauses binaires améliore considérablement l'efficacité des algorithmes de résolution. On sait aussi que les clauses de Horn sont à la base des langages de programmation logique, etc. On retrouve ensuite les formules Horn-renommables (des formules qui se réduisent à Horn par un renommage adéquat des littéraux) et les q-Horn (une généralisation de Horn-SAT et 2-SAT). D'autres restrictions polynômiales ont été exhibées soit par l'introduction de différentes formes de hiérarchies d'instances (e.g. hiérarchie de G. Gallo et M.G. Scutellà [Gallo & Scutellà 1988], hiérarchies de Dalal-Etherington [Dalal & Etherington 1992]), soit en restreignant le nombre d'occurrences des variables (e.g. r,r-SAT de Tovey[3] [Tovey 1984, Dubois 1990], formules dont les variables apparaissent au plus deux fois [Tovey 1984]). D'autres classes ont été obtenues grâce à des formulations en programmation linéaire (e.g. les formules équilibrées [Conforti & Cornuéjols 1992], généralisation de clauses de Horn [Chandru & Hooker 1990]). Plus récemment, en se basant sur la constatation qu'une utilisation appropriée de la procédure de Davis et Putnam [Davis *et al.* 1962] permet de récupérer une bonne partie des classes polynômiales connues, d'autres généralisations ont été obtenues, la reconnaissance dans ces cas est basée sur un algorithme plutôt que sur les propriétés de la formule (e.g. les travaux d'A. RAUZY et R. GÉNISSON [Rauzy 1994, Rauzy 1995b, Génisson & Rauzy 1996], la classe SLUR (Single-Lookahead Unit Resolution) [Schlipf *et al.* 1995]).

2.1.2 Algorithmes de résolution

De nombreux algorithmes ont été proposés pour résoudre SAT. Il serait présomptueux de prétendre réaliser un inventaire complet de ces méthodes. Les principes à la base de ces méthodes sont très variés : résolution [Robinson 1965, Kowalski & Kuehner 1971, Loveland 1978], réécriture [Boole 1854, Shannon 1938], énumération [Davis *et al.* 1962, Jeroslow & Wang 1990], comptage du nombre de solutions [Iwama 1989], recherche locale [Selman *et al.* 1992], programmation linéaire en nombre entiers [C.E. Blair & Lowe 1986, Hooker 1988, Bennaceur & Plateau 1992, Bennaceur & Plateau 1995], diagrammes binaires de décision [Akers 1978, Bryant 1992, Uribe & Stickel 1994], algorithmes génétiques et évolutionnistes [Hao 1995], etc. On peut distinguer parmi ces différentes méthodes deux catégories : les algorithmes complets et les algorithmes incomplets. Un algorithme pour SAT est *incomplet*, si pour toute formule pour laquelle la réponse est « oui », celle-ci est satisfiable, et *complet* si en plus la réponse « oui » (resp. « non ») est retournée (en un temps fini) pour toute formule satisfiable (resp. insatisfiable).

J'ai essentiellement travaillé et/ou utilisé les algorithmes basés sur la résolution, l'énumération et la recherche locale. Sans viser à être exhaustif, dans la suite, je limite la présentation aux algorithmes utilisant ces trois principes. Les techniques à base d'énumération et de recherche locale restent de loin les plus efficaces en pratique. En outre, je pense que la résolution, par son histoire, sa complémentarité aux autres approches et la puissance de plus en plus croissante des machines jouera un rôle encore plus important dans la prochaine génération d'algorithmes.

3. r,s-SAT désigne l'ensemble des formules où les clauses ont exactement r littéraux et chaque variable admet au plus s occurrences.

2.1.2.1 Résolution

Le principe (ou règle) de résolution consiste à produire à partir de deux clauses $C_1 = (l \vee V)$ et $C_2 = (\neg l \vee W)$, une clause (appelée *résolvante*) $Res_l(C_1, C_2) = (V \vee W)$. La *résolution* consiste à répéter l'application de principe jusqu'à la génération d'une clause vide (auquel cas la formule est insatisfiable). On appelle réfutation, toute suite de clauses générées par ce processus, la dernière étant une clause vide [4]. Une réfutation peut être représentée par un *arbre binaire de résolution* dont la racine est la clause vide, tout nœud interne est étiqueté par la résolvante produite à partir de ses deux fils, les feuilles sont étiquetées par des clauses de la formule.
La résolution dont l'origine remonte à Blake [Blake 1937] (voir aussi une variante de la résolution, la version originale de la procédure de Davis et Putnam [Davis & Putnam 1960]) est souvent attribuée à Robinson [Robinson 1965] qui a proposé une procédure de preuve en calcul des prédicats. Cette procédure permet de combiner le principe de résolution et d'unification [Robinson 1965]. Cette méthode est relativement inefficace car le nombre de résolvantes à engendrer est souvent important. De nombreuses stratégies ont été proposées afin de limiter le nombre de clauses produites ; parmi elles, citons la résolution linéaire [5] et ses diverses variantes (SL-résolution [Kowalski & Kuehner 1971], SLRI (SL-résolution avec Remontée des Impasses) [Cubbada & Mouseigne 1988]), la résolution régulière [6] [Tseitin 1968], la résolution étendue [7] [Tseitin 1968], ..., la procédure de Davis et Putnam (DP) [Davis & Putnam 1960]. La version originale de DP est une variante de la résolution, elle effectue l'élimination successive des variables, en générant pour chaque variable choisie par une heuristique [8], l'ensemble des résolvantes possibles et en éliminant les clauses mentionnant une telle variable. A chaque étape, le sous-problème généré contient une variable en moins, mais un nombre quadratique de clauses supplémentaires. DP exploite en plus les littéraux purs et unitaires.

2.1.2.2 Énumération

L'idée de base des algorithmes énumératifs est la construction d'un arbre binaire de recherche où chaque nœud représente une instanciation partielle des variables. À chaque nœud de l'arbre, on procède généralement à un filtrage (simplification) de la formule courante. Le choix de la prochaine variable à instancier fait l'objet en général d'une heuristique. Un exemple typique est la procédure de Davis, Logemann et Loveland (communément appelée DPL) [Davis *et al.* 1962] : la simplification s'appuie sur la propagation des littéraux purs et des mono-littéraux ; le choix de la prochaine variable à affecter est fait suivant une heuristique (e.g. choix de la variable figurant le plus souvent dans les clauses les plus courtes). La différence entre les deux procédures DP et DPL réside dans le fait que la première procède par élimination de variables en remplaçant le problème initial par un sous-problème plus large, alors que la seconde procède par séparation [9], en remplaçant le problème original par deux sous-problèmes de taille plus réduite. Ces deux procédures admettent des comportements différents et sont incomparables du point de vue de l'analyse de la complexité [Goldberg 1979b, Dechter & Rish 1994]. La procédure DPL est plus souvent implantée et utilisée pour résoudre SAT que la version originale DP ; car la règle d'élimination de variables admet plusieurs inconvénients : elle est plus difficile à programmer que la règle de séparation ; elle tend très rapidement à augmenter le nombre de clauses ; elle génère de nombreuses clauses redondantes ou sous-sommées [10].

4. La résolution est complète pour la réfutation.
5. La résolution est dite linéaire, si à chaque étape (sauf la première), une résolvante est produite à partir de la résolvante courante et d'une résolvante précédente ou d'une clause de la formule.
6. La résolution est dite régulière, si pour tout chemin dans l'arbre de résolution, une variable est éliminée au plus une fois.
7. La résolution étendue, utilise en plus de la règle de résolution, le principe *d'extension*, qui consiste à introduire de nouvelles variables représentant des formules intermédiaire (lemmes).
8. Dans la version originale de DP, la prochaine variable à éliminer est choisie dans la première clause de longueur minimale.
9. Soit $\mathcal{F}$ un ensemble de clauses et p une variable de $\mathcal{F}$, $\mathcal{F}$ est satisfiable si et seulement si $\mathcal{F}(p)$ ou $\mathcal{F}(\neg p)$ est satisfiable.
10. Une clauses C sous-somme C' si et seulement si $C \subset C'$.

Ci-dessous, une description détaillée de la procédure DPL :

Algorithme 2.1 DPL

```
Function DP : boolean
Input  : un ensemble 𝓕 de clauses
Output : true si 𝓕 est consistant, false sinon
Begin
    𝓕=Simplifier( 𝓕) ;                %% propagation des littéraux purs et unitaires
   if ( 𝓕 contient la clause vide) then return false ;
   elif ( 𝓕==∅) then return true ;
   else
     l=Heuristique_de_branchement( 𝓕) ;
     return DP( 𝓕(l)) || DP( 𝓕(¬ l)) ;
   fi
End
```

Par sa simplicité, la procédure DPL est l'une des plus utilisée pour résoudre SAT et est actuellement la plus performante en pratique. Les deux points clés de cette procédure sont la simplification et l'heuristique de branchement utilisée. La première permet de transformer la formule en une formule plus simple équivalente pour SAT ; la seconde a une incidence directe sur la taille de l'arbre de recherche. De nombreuses améliorations lui ont été apportées, elles concernent généralement un (ou plusieurs) des points suivants :

1. simplification de la formule ;
2. choix de la prochaine variable à affecter ;
3. traitement des échecs.

On distingue en général deux types d'approches, les approches prospectives (de type « *look-ahead* ») qui permettent de détecter de futures situations d'échec, et les approches rétrospectives (de type « *look-back* ») qui essayent d'apprendre à partir des situations d'échec. Ce qui correspond respectivement aux améliorations apportées à l'étape de simplification et au traitement des échecs.

Simplifications : Une technique de simplification est intéressante si elle permet non seulement de réduire le nombre de nœuds de l'arbre de recherche, mais également le temps de calcul. En plus de la propagation des littéraux purs et unitaires, de nombreuses techniques de simplification ont été introduites et utilisées, soit comme pré-traitement à la formule, soit au cours de la recherche. Parmi les plus intéressantes, on peut citer :

– la *résolution restreinte* qui consiste à ajouter des résolvantes de taille bornée. Cette technique est très souvent utilisée comme pré-traitement à la formule et permet en particulier de détecter certaines inconsistances locales, et de mieux traiter certaines classes de formules [Boufkhad 1996]. Différentes formes de résolution restreinte plus ou moins complexes ont été exploitées ces dernières années [Billionnet & Sutter 1992, Génisson & Siegel 1994, Van Gelder & Tsuji 1996], etc.
– Les *traitements locaux par propagation unitaire* ont été introduits dans CSAT par Olivier Dubois et al. [Dubois *et al.* 1996]. Ces traitements, appliqués pendant la résolution, permettent de produire par propagation unitaire des littéraux impliqués ou de détecter des inconsistances locales. Cette idée d'exploiter au maximum la propagation unitaire a été utilisée dans la plupart des implémentations performantes actuelles de la procédure DPL (Tableau, POSIT, Satz, etc.).

– L'*exploitation des classes polynômiales* au cours de la résolution a fait l'objet de nombreux travaux. Parmi les restrictions polynômiales connues, 2-SAT et Horn SAT restent les plus exploitées dans le cadre général [Gallo & Urbani 1989, Buno & Buning 1993, E. Boros & Kogan 1994, Larrabee 1992]. Ceci s'explique an partie par le fait que dans de nombreux problèmes réels, certaines contraintes peuvent être codées par des clauses binaires ou des clauses de Horn.

Heuristiques : Du fait de la relation forte entre l'ordre d'affectation des variables et la taille de l'arbre de recherche développé, une « bonne » heuristique de branchement est déterminante pour l'efficacité d'un algorithme de recherche. De nombreuses heuristiques ont été proposées dans la littérature, les plus populaires aujourd'hui ne sont que des variantes de celles suggérées dans [Davis *et al.* 1962, Goldberg 1979b, Jeroslow & Wang 1990] et décrites par PRETOLANI [Pretolani 1993] comme l'heuristique « *Moms* » : choix de la variable ayant le maximum d'occurrences dans les clauses les plus courtes. La première amélioration significative a été proposée dans [Dubois *et al.* 1996], elle se base sur une nouvelle fonction de pondération des clauses et sur l'ajout d'un facteur additionnel pour équilibrer les arbres de recherche. On peut trouver ensuite d'autres variantes, obtenues en exploitant par exemple le traitement local par propagation unitaire [Freeman 1995, Li & Anbulagan 1997], en estimant à l'avance l'effet de la propagation unitaire [Boufkhad 1996] ou encore en considérant les occurrences dans les clauses non Horn [Zhang 1997, Crawford & Auton 1996].

Traitement des échecs : En analysant les situations d'échec, les approches rétrospectives permettent de réaliser un retour-arrière non chronologique. Ce type d'approches a été étudié et appliqué dans de nombreux domaines de l'IA, comme dans les systèmes de maintien de vérité *ATMS* [McAllester 1980, Stallman & Sussman 1977], les problèmes de satisfaction de contraintes [Dechter 1989, Ginsberg 1993, Schiex & Verfaillie 1993] et en démonstration automatique [Bruynooghe 1980]. Elles se distinguent essentiellement par les techniques utilisées pour analyser les échecs et pour éviter de rencontrer les mêmes situations au cours de la recherche. Dans le cadre de SAT, les deux meilleures techniques sont GRASP [Silva & Sakallah 1996] et Relsat [Bayardo Jr. & Schrag 1997]. L'efficacité de ce type d'approches dépend à la fois des problèmes traités et des heuristiques de branchement utilisées. Une bonne heuristique de branchement tend à diminuer la hauteur du saut ou du retour-arrière. L'expérience montre que ce type d'approches a pour effet de corriger les conséquences d'un mauvais choix de variables; ce qui n'est pas exclu lorsqu'on utilise une heuristique, très souvent basée sur des arguments syntaxiques. On peut ranger aussi dans ce cadre l'évaluation sémantique d'OXUSOFF et RAUZY [Oxusoff & Rauzy 1989] et la généralisation proposée par Thierry CASTELL [Castell 1996]. Ces différentes techniques obtiennent leur meilleure performance sur certaines classes de problèmes réels. Il est inutile de les appliquer pour résoudre des instances aléatoires.

2.1.2.3 Recherche locale

Les méthodes considérées ici, contrairement aux algorithmes énumératifs, considèrent des « configurations », c'est-à-dire des instanciations complètes de toutes les variables de la formule. Au départ, ce sont des méthodes d'optimisation (on peut par exemple chercher à maximiser le nombre de clauses satisfaites). Sauf adaptation, ces méthodes sont incomplètes, et ne garantissent donc pas l'obtention d'une solution. En particulier, elles ne permettent pas de prouver qu'une instance est insatisfiable.

Le principe de base des méthodes de recherche locale consiste à se déplacer judicieusement dans l'espace des configurations en améliorant la configuration courante. On peut grossièrement résumer le calcul d'une solution par une méthode de recherche locale de la manière suivante :

Algorithme 2.2 RL

```
Function RL : boolean
Input  :  𝓕 un ensemble de clauses, deux entiers : nbEssais et nbDeplacements ;
Output : true si  𝓕 admet un modèle, false s'il est impossible de conclure ;
Begin
   for i from 1 to nbEssais
   do
       I = interprétationInitiale( F) ;           %% e.g. générée aléatoirement
       for j from 1 to nbDeplacements
       do
          if (I est un modèle de  F)
          then
              return true ;
          else
             do
                P = meilleursVoisins(I,  F) ;          %% e.g. « min-conflict »
                v = choix(P) ;      %% e.g. choisir une variable au hasard parmi P
             done
             I = inverser(I,v) ;                  %% inverser la valeur de v dans I
          fi
       done
   done
   return false ;                                               %% modèle non trouvé
End
```

En général, les algorithmes de recherche locale sont incomplets ; ils ne permettent donc pas de prouver l'inconsistance et aboutissent généralement a un optimum local. On savait que ce type de méthodes sont très efficaces pour résoudre des problèmes d'optimisation combinatoire. La surprise est venue de ce qu'elles sont aussi très efficaces pour traiter SAT, où une solution exacte est demandée. En effet, au début des années 90, les raffinement proposés par [Minton *et al.* 1990, Chabrier *et al.* 1991, Sosic & Gu 1991], et ensuite par [Selman *et al.* 1992] ont montré leur efficacité pour résoudre certains problèmes difficiles qu'aucune méthode complète n'est capable de traiter encore à l'heure actuelle[11]. Les méthodes de recherche locale ont été appliquées avec succès sur des instances de très grande taille du problème des reines et sur d'autres problèmes structurés, alors que GSAT de B. SELMAN a été appliqué avec succès sur des instances aléatoires difficiles. Ce qui explique à mon avis, le large écho accordé dans la littérature à GSAT comparativement aux autres méthodes. Cette efficacité remarquable a motivé la mise au point de nouvelles variantes [Morris 1993, Gent & Walsh 1993, Selman *et al.* 1994, Cha & Iwama 1995, Hao & Tétard 1996, McAllester *et al.* 1997]. Leurs différences se situent essentiellement dans la fonction de sélection de la prochaine variable à réparer[12] et dans la stratégie utilisée pour s'échapper des minima locaux.

2.1.3 Instances de SAT

L'évaluation d'un algorithme est une étape aussi indispensable que sa mise en œuvre. Comme il semble improbable de trouver un jour un algorithme en temps polynômial pour SAT, déterminer ce qu'est un « bon »

11. Voir [Franco *et al.* 1997], pour une note historique (discutable) sur les techniques de recherche locale, En effet, on peut la compléter par les travaux de HANSSEN et JAUMARD sur Max-Sat [Hansen & Jaumard 1990], ceux de CHABRIER et al. [Chabrier *et al.* 1991] et enfin par une méthode plus ancienne appelée « *Inversion method* » et proposée dans [Dunham & Wang 1976].

12. Réparer une variable consiste à inverser sa valeur de vérité dans l'affectation courante, on utilise aussi le terme « *flipper* ».

algorithme est en soi une question très difficile. En effet, d'un point de vue théorique, l'analyse de sa complexité en moyenne est souvent difficile à mettre en œuvre, alors que d'un point de vue pratique, nous nous heurtons à la difficulté d'établir un jeu de tests «significatifs ».

Cependant l'évaluation expérimentale présente le mérite de permettre l'identification des propriétés des algorithmes et des problèmes traités, qui jusqu'à présent n'ont pas pu être analysés de manière rigoureuse. La recherche d'instances difficiles a pour objectifs de caractériser les limitations des algorithmes et de montrer là où des investigations supplémentaires sont nécessaires.

On peut regrouper les instances utilisées pour évaluer les algorithmes en trois catégories :

1. *problèmes structurés* : on retrouve dans cette catégorie, les puzzles logiques (e.g. problèmes de Lewis Caroll [Carrol 1966], les reines, le problème du zèbre, etc.), le problème des pigeons et les formules de Tseitin [Tseitin 1968], les problèmes de Ramsey [Ramsey 1930], coloriage de graphes, etc.
2. *problèmes aléatoires* : parmi les modèles de génération aléatoire proposés, deux modèles ont été principalement étudiés :
   – *modèle à densité fixée* [Goldberg 1979b] : étant donnés un nombre n de variables, un nombre m de clauses, et une probabilité p fixée. Chaque clause est générée indépendamment, en incluant avec une probabilité p chacun des $2n$ littéraux. Ce modèle génère des instances faciles en moyenne [Goldberg 1979b]. Franco et Paul [Franco & Paul 1983] ont montré que les résultats obtenus par GOLDBERG sont une conséquence directe du modèle utilisé. Ce qui a diminué considérablement la portée de ce modèle.
   – *modèle k-SAT (clause de taille fixée)* : étant donnés un nombre n de variables, un nombre m de clauses, et une longueur k des clauses fixée. Chaque clause est générée indépendamment, en tirant au hasard k variables parmi les n variables et en négativant chaque variable avec une probabilité de $\frac{1}{2}$. Un des résultats les plus importants de ces dernières années est le « phénomène de seuil » mis en évidence sur les données $k - SAT$ aléatoires [AI 1996]. Les expérimentations ont montré une séparation aiguë entre données satisfiables et données insatisfiables (la courbe de probabilité de satisfiabilité passe brutalement de 1 vers 0, lorsque le rapport du nombre de clauses sur nombre de variables augmente, le nombre de variables étant fixé). Le seuil est défini par la limite du rapport m/n quand n devient infini. Ce modèle permet pour la première fois de caractériser des instances SAT difficiles à résoudre en moyenne par toutes les méthodes développées. Ces instances se situent au seuil. Pour 3-SAT par exemple, la valeur expérimentale du seuil (la valeur du ratio m/n telle que la probabilité de tirer une instance satisfiable est égale à $\frac{1}{2}$) est proche de $4, 25$. Ce phénomène de seuil observé expérimentalement et indépendamment par plusieurs auteurs [Dubois & Carlier 1991, Cheeseman *et al.* 1991, Mitchell *et al.* 1992, André 1993, Larrabee & Tusji 1993, Crawford & Auton 1993] a permis le développement de méthodes de résolution du problème SAT plus efficaces. Pour des instances $3 - SAT$ générées au seuil les meilleures méthodes complètes ne peuvent résoudre des instances au-delà de 500 variables [13], alors que les méthodes incomplètes de type recherche locale sont capables de résoudre des instances satisfiables au seuil au-delà de 2000 variables. Sur le plan théorique, le calcul des seuils pour chaque classe $k - SAT$ en même temps que la preuve de l'existence du phénomène sont activement recherchés [Chvátal & Reed 1992, Goerdt 1992, Kamath *et al.* 1994, Frieze & Suen 1996, Dubois & Boufkhad 1997, Dubois 2000]. L'intérêt de ce modèle est aujourd'hui très largement admis. Cependant, les instances générées par ce modèle sont loin d'être proches des instances issues d'applications réelles. On peut aussi citer d'autres modèles de génération qui ont été mis au point soit dans le but de générer des instances encore plus difficile , soit pour obtenir des instances satisfiables difficiles pour la recherche locale, ou encore "plus proches" des instances qu'on trouve dans le monde réel (cf. section 2.5).
3. *problèmes réels* : Les instances issues d'applications réelles admettent leurs propres caractéristiques, qui peuvent être exploitées en vue d'un traitement efficace d'une application particulière. Ce qui peut

13. Prouver l'insatisfiablité des instances à 700 variables au seuil, est proposé comme un challenge dans [Selman *et al.* 1997].

donner ensuite des indications importantes pour une généralisation à la résolution d'autres domaines d'applications. Ces dernières années ont vu une effervescence particulière autour de la résolution d'applications réelles (e.g. vérification de circuits, planification, vérification de modèles (« *model checking* »), cryptographie, etc.). Cette tendance a été stimulée, d'une part par les progrès récents obtenus sur la résolution pratique de SAT, et d'autre part par les deux compétitions organisées ces dernières années. La première a été organisée en 1993 par le centre de recherche en mathématiques discrètes de Rutgers (DIMACS), la seconde par l'école d'aéronautique de Pékin en 1996. Ce qui a permis notamment de définir un format standard pour coder les instances SAT (format DIMACS) facilitant ainsi la mise en place de librairies de problèmes (DIMACS [Dimacs 1993], SATLIB [Hoos & Stützle 2000]).

2.1.4 Analyse de complexité

De nombreux résultats de complexité ont été obtenus pour certaines des méthodes citées précédemment. On a, par exemple pour la résolution, cherché à améliorer le meilleur minorant connu de la taille des preuves [Tseitin 1968, Galil 1977, Krishnamurthy 1982, Haken 1985, Urquhart 1987]. Ces différents résultats ont été obtenus sur des formules très fortement structurées comme celles codant le problème des pigeons [14], les formules de Tseitin et d'Urquahrt [15], les formules codant des instances du théorème de Ramsey [16] [Ramsey 1930], etc. Certaines de ces formules peuvent être résolues efficacement, en utilisant par exemple la résolution étendue (e.g. [Cook 1976]) ou la résolution avec symétries [Krishnamurthy 1985] (cf. section 2.2.1).

On peut également mentionner les résultats de GOLDBERG et al. [Goldberg 1979a, Goldberg *et al.* 1983] et de PURDOM [Purdom 1990] sur le modèle à densité fixé, montrant que la procédure de Davis et Putnam admet en moyenne un comportement polynômial, les résultats dans le pire cas obtenus par de nombreux auteurs pour les algorithmes énumératifs, de type DPL [Monien & Speckenmeyer 1985, Schiermeyer 1983, Schiermeyer 1996, Kullmann 1999a], ou encore le résultat important de Vašek CHVATAL et Endre SZEMEREDI [Chvátal & Szemerédi 1988] montrant que la résolution est exponentielle sur des instances k-SAT aléatoires. D'autres résultats plus récents ont été obtenus par VANGELDER [van Gelder 1999] et DANTSIN et al. [Dantsin *et al.* 2000].

Les résultats de complexité concernant les techniques à base de recherche locale sont plus difficiles à obtenir, on peut trouver une analyse dans le pire cas pour l'algorithme GSAT dans [Hirsch 2000].

2.2 Exploitation de propriétés structurelles

Pour certaines classes de problèmes réels ou structurés, une des voies pour réduire considérablement l'espace de recherche (cette réduction peut être exponentielle dans certains cas) est sans aucun doute la prise en compte des informations structurelles de la formule considérée. Les algorithmes cités précédemment ignorent pour la plupart ce type d'informations qui sont souvent à l'origine de la difficulté de résolution. Il est à noter que, selon le formalisme utilisé pour coder un problème, ces informations peuvent être plus ou moins apparentes et faciles à exploiter. On peut citer, par exemple, un domaine où la structure du problème joue un rôle considérable, il s'agit des problèmes de satisfaction de contraintes (CSP). En effet,

14. « *Pigeon-Hole problem* » en anglais et « *tiroirs et chaussettes* » en français. Le problème consiste à placer n pigeons dans $n-1$ pigeonniers, avec un et un seul pigeon par pigeonnier.

15. Ces formules se modélisent sous forme d'une conjonction de chaînes d'équivalences (Biconditional Normal Form) et donc résolvables en temps polynômial [Dunham & Wang 1976].

16. Pour donner une idée du théorème de Ramsey, partons de la division par excès $\frac{P}{k}$ = le plus petit entier p qui, multiplié par k, donne $p \times k \geq P$. D'une manière plus intuitive, si nous répartissons P objets dans k tiroirs, l'un au moins des tiroirs comprend au moins p objets.

la représentation en terme de (hyper)graphes d'un CSP offre un avantage certain. Citons, par exemple, les travaux développés autour des techniques de décomposition de CSP [Jégou 1993, Chmeiss 1996], la mise en évidence de classes traitables [Freuder 1982, Dechter 1990, Dechter & Pearl 1989], etc.

Les arguments développés ci-dessus expriment ma démarche scientifique. En effet, je me suis intéressé aux aspects liés au codage, après avoir travaillé sur l'exploitation des propriétés structurelles.

Les travaux autour de ces aspects ont été moins nombreux dans le cadre du problème SAT. On peut constater cependant un regain d'intérêt autour de ce type d'approches, motivé principalement par les applications réelles qui commencent à être résolues en utilisant les techniques de résolution de SAT. On peut imaginer différentes informations structurelles, je me suis intéressé particulièrement à deux types de structures : les symétries et les équivalences.

2.2.1 Symétries

La notion de symétrie est très présente dans la vie courante et dans de nombreux domaines comme la physique, la chimie, les mathématiques, etc. Cette notion, permet par exemple de simplifier certaines preuves mathématiques. Dans le même esprit, notre objectif premier est de voir dans quelle mesure il est possible d'exploiter ces régularités présentes dans de nombreux problèmes réels et structurés (les formules logiques qui les représentent restent invariantes sous certaines permutations de noms de variables) pour améliorer les performances des algorithmes de résolution.

En outre, les formules ingénieuses construites par Tseitin [Tseitin 1968] (le problème des pigeons, etc.) pour établir la complexité de la résolution, admettent des preuves polynômiales en augmentant la résolution soit par la règle d'extension (résolution étendue) proposée par le même auteur, soit par la règle de symétrie proposée par B. KRISHANMURTY [Krishnamurthy 1985]. Ce dernier a affirmé que pour la résolution, tout nouveau résultat de complexité passe nécessairement par l'utilisation du principe d'extension et de symétrie. Au début des années 90, Pierre SIEGEL [Siegel 1990] m'a expliqué les motivations liées à l'étude des symétries en calcul propositionnel. En effet, fort de l'idée que, lors de la résolution de problèmes difficiles, on peut tomber sur un sous-problème de type « *Pigeon-Hole* », cette étude peut aussi servir à mieux résoudre des problèmes qui peuvent ne pas présenter des symétries apparentes au départ.

Cette affirmation qui peut paraître au premier abord utopiste, je l'ai trouvée récemment dans le papier de Douglas D. EDWARDS [Edwards 1996] qui citait Bart SELMAN : « ..., Bart Selman (personal communication) believes that hidden pigeon-hole substructure is a common source of intractability in realistic UNSAT problems ». En se basant notamment sur les récents travaux de Craword et Baker [Crawford & Baker 1994] sur la résolution de problèmes d'ordonnancement (« *scheduling problems* »), des arguments ont été avancés pour justifier cette forte conjecture.

Dans tous les cas, la prise en compte des symétries par une méthode de résolution réclame un algorithme de *détection* de symétries d'une part, et une technique d'*exploitation* des symétries trouvées, d'autre part.

Une partie de notre travail est axée sur la détection des symétries et leur utilisation dans les algorithmes de résolution. Dans un premier temps, ce travail a abouti à l'implémentation d'un algorithme de détection des symétries dans une formule logique donnée sous Forme Normale Conjonctive **[Benhamou & Saïs 1991, Benhamou & Saïs 1992, Benhamou *et al.* 1992b]**. Du point de vue de la complexité, nous avons montré que le problème de la détection des symétries est polynômialement équivalent au problème de l'isomorphisme de graphes **[Saïs 1993]**. Nous avons ensuite introduit les symétries dans différentes méthodes : SL-Résolution, procédure DPL et Evaluation Sémantique. Les motivations précédemment développées nous ont conduits à opter pour une exploitation dynamique (au cours de la recherche) des symétries. La détection des symétries étant un problème difficile, nous avons choisi d'utiliser une version incomplète de l'algorithme de détection. Des résultats meilleurs ont été obtenus sur la plupart des problèmes testés, en particulier sur les problèmes combinatoires du type recherche de nombres de Ramsey, coloriage de graphes, etc. Une preuve de complexité linéaire a été donnée pour le problème des pigeons. En outre, pour la première fois, le nombre de Ramsey $R(3,3,3)$ [17] a été trouvé par une méthode de déduction. Grâce à l'utilisation de conditions nécessaires à la présence de symétries, d'une stratégie intelligente et de structures de données adaptées, la recherche de symétries n'est que très peu coûteuse en pratique (surcoût de 10% environ quand le problème traité ne contient pas de symétries) **[Benhamou & Saïs 1992, Benhamou & Saïs 1994]**.

Par ailleurs, nous avons été conduits à introduire un concept particulier de symétrie qualifiée de "forte" qui vérifie la condition supplémentaire suivante : quand les k premiers littéraux d'un cycle de symétries C (ensemble de littéraux symétriques deux à deux) sont interprétés à $vrai$, le sous-ensemble de littéraux de C restant après chaque étape de l'interprétation est aussi un cycle de symétries. C est alors un cycle dit "k-fortement symétrique" ("fortement symétrique" lorsque k est égal au nombre de littéraux de C). Cette notion de symétrie forte permet la généralisation des propriétés des symétries simples. En effet, selon le théorème des symétries simples (développé dans ma thèse), si deux littéraux a et b d'une formule $\mathcal{F}$ appartiennent à un même cycle de symétries C, alors, $\mathcal{F} \wedge a$ est satisfiable si et seulement si $\mathcal{F} \wedge b$ est satisfiable. Généralisation : si A est un sous-ensemble de littéraux d'un cycle fortement symétrique C de $\mathcal{F}$, alors $\mathcal{F} \wedge A$ est satisfiable si et seulement si $\mathcal{F} \wedge B$ est satisfiable, pour tout sous-ensemble B de C. Cette nouvelle notion de symétrie est automatiquement utilisable dans le formalisme des formules de cardinalité ; elle permet la production immédiate d'une information supplémentaire précisément caractérisable par une formule de cardinalité (cf. section 2.3.1), donc de raccourcir considérablement la preuve de certains problèmes **[Benhamou *et al.* 1992b, Saïs 1993]**. En effet, si $\mathcal{F} \wedge A$ est insatisfiable, alors on peut déduire l'information suivante : "au plus $|A| - 1$ littéraux de $vrai$ parmi ceux de C".

J'ai effectué une évaluation expérimentale de la procédure DPL avec et sans symétries sur certaines classes de problèmes proposés lors du challenge international organisé par DIMACS. Des résultats intéressants ont été obtenus sur de nombreux problèmes, en particulier sur les formules de *Tseitin* et d'*Urquhart* intraitables par les algorithmes classiques. Notre méthode a été citée lors de ce challenge comme la seule à avoir été capable de résoudre de telles formules **[Saïs 1994b]**.

L'étude des symétries (ou d'autres formes plus faibles) a fait l'objet de nombreux travaux dans différents domaines comme la logique du premier ordre [Crawford 1992, Zhang & Zhang 1995, Slaney *et al.* 1993, Slaney 1994], les problèmes de satisfaction de contraintes [Benhamou 1994, Ellman 1993, Puget 1993, Benson & Freuder 1992], et bien évidement en calcul propositionnel [Aguirre 1992, Chabrier *et al.* 1995, Belleannée & Vorc'h 1994, Boy de la Tour & Demri 1995, Crawford *et al.* 1996, Chabrier 1997]. Je citerai d'une part les travaux de J. CRAWFORD sur les symétries en logique du premier ordre [Crawford 1992], que nous avons découverts pour la première fois lors du « *workshop on tractable reasoning, AAAI'92* » auquel nous avons participé. Dans son papier, CRAWFORD a obtenu un résultat particulièrement intéressant autour de l'utilisation du théorème de Polya [Polya & Read 1987] pour compter le nombre d'interprétations non isomorphes [Crawford 1992]. Dans [Crawford *et al.* 1996], il est proposé une technique d'élimination de symétries par ajout de nouvelles contraintes. Cette technique, peut être utilisée comme pré-traitement de la formule. Cependant, le nombre de contraintes à ajouter peut être prohibitif. D'autre part, un langage de très haut niveau a été mis au point au *LIRSIA* (Université de Bourgogne, Dijon), supportant une recherche locale complète appelé SCORE(FD/B) [Chabrier *et al.* 1996, Chabrier 1997]. Ce langage de représentation fournit des primitives, clauses de cardinalité (cf. section 2.3.1) et structures de contrôle, exploitées statiquement par le compilateur pour déterminer des propriétés des problèmes, par exemple les symétries.

17. $R(3,3,3) = 17$, le plus petit nombre de sommets à partir duquel il est impossible de colorier les arêtes d'un graphe complet avec 3 couleurs sans obtenir un triangle monochromatique. Un graphe à 16 sommets admet un tel coloriage.

2.2.2 Équivalences

L'opérateur d'équivalence ($\Leftrightarrow$) joue un rôle important en logique propositionnelle. En effet, en plus de son utilisation en tant qu'opérateur pour définir des fonctions booléennes quelconques, il est à la base du principe d'extension de *Tseitin* où il est utilisé pour représenter des formules intermédiaires dont la manipulation permet de raccourcir les preuves. Il est aussi très utilisé pour construire des formules très difficiles à la fois pour les algorithmes complets et incomplets. On peut citer les formules de *Tseitin* pour lesquelles la résolution est exponentielle, les formules codant le problème de parité (instances « Parity 32 » de DIMACS) très difficiles à la fois pour les algorithmes complets et incomplets, ou encore les formules difficiles pour la recherche locale, construites par Edward *Hirsch* [Hirsch 2000]. On retrouve ces équivalences dans de nombreux problèmes réels (problèmes de circuit, de model checking, de cryptographie, etc.).

Une des propriétés les plus intéréssantes est le fait que si on a deux littéraux équivalents ($a \Leftrightarrow b$), on peut substituer le littéral a par b partout dans la formule, en gardant une équivalence du point de vue de la satisfiabilité. Cela permet de réduire le nombre de variables et de produire éventuellement de nouveaux littéraux ou encore de détecter des contradictions.

Il faut aussi noter que parmi les formules difficiles citées précédemment, certaines peuvent être entièrement (ou partiellement) codées sous forme d'une conjonction de chaînes d'équivalences (Biconditional Normal Form (BNF)). Les formules de ce type peuvent être résolues en temps polynômial [Dunham & Wang 1976].

Ces remarques justifient à elles seules l'utilisation spécifique des équivalences pour améliorer la résolution de certaines classes de problèmes.

Comme pour les symétries, la prise en compte des équivalences pose le problème de leur *détection*, de leur *utilisation* et du *choix du formalisme* permettant une expression plus naturelle de ce type de structure.

En collaboration avec Laure BRISOUX et Éric GRÉGOIRE, nous avons opté pour une approche sémantique, par propagation unitaire. En effet, pour une variable donnée x, en analysant les littéraux produits par propagation unitaire de $\mathcal{F} \wedge x$ et de $\mathcal{F} \wedge \neg x$, on peut détecter des inconsistances locales et produire non seulement des littéraux, mais également des équivalences entre littéraux. Ce qui permet de généraliser les traitements locaux par propagation unitaire utilisés traditionnellement par de nombreux auteurs. Les littéraux équivalents peuvent ensuite être utilisés en effectuant des substitutions complètes ou partielles. Plus précisément, si $a \Leftrightarrow b$ dans $\mathcal{F}$, alors tout en obtenant une formule équivalente pour SAT, substituer a par b et $\neg a$ par $\neg b$ dans $\mathcal{F}$, ou encore limiter cette substitution de a par b uniquement aux clauses contenant à la fois a (ou $\neg a$) et b (ou $\neg b$). Ce qui revient dans ce dernier cas à satisfaire ou à raccourcir les clauses. Une description de ce travail peut être trouvée dans la thèse de Laure BRISOUX **[Brisoux-Devendeville 1999]**. Daniel LE BERRE propose d'étendre ce processus de détection en effectuant une double propagation unitaire, ce qui permettra de détecter des informations additionnelles. Cette généralisation pose des problèmes de coût ; une collaboration est actuellement en cours sur ce sujet.

Une autre voie consiste à faire des abstractions basées sur la syntaxe. En d'autres termes, reconnaître les chaînes d'équivalences de manière syntaxique à partir de la formule CNF. Une reconnaissance d'équivalences binaires simples a été utilisée par BOUFKHAD [Boufkhad 1996] comme pré-traitement de la formule considérée. Li CHU-MIN dans son papier [Chu-Min 2000] effectue à tous les nœuds de l'arbre de recherche une reconnaissance syntaxique de telles chaînes (limitée à des chaînes de longueur ≤ 3), et utilise un certain nombre de règles de déduction pour raisonner sur de telles formules. Sachant que la classe de formules BNF est stable par adjonction de littéraux unitaires, il est plus judicieux de faire cette reconnaissance sur la formule de départ et d'appliquer ensuite l'algorithme proposé par WANG [Dunham & Wang 1976] pour simplifier la formule.

L'un des inconvénients de la recherche locale, est certainement la non prise en compte de la structure de la formule (e.g. longueur des clauses, les équivalences). Dans **[Brisoux *et al.* 2000]**, nous proposons une utilisation du traitement local par propagation unitaire dans le cadre de la recherche locale. Ce processus de propagation unitaire est effectué d'une part comme pré-traitement de la formule, mais également pour exhiber les dépendances entre littéraux [18] qui sont ensuite utilisées par un algorithme de recherche locale. Signalons que la notion d'équivalence entre littéraux est un cas particulier de la notion de définissabilité définie dans [Lang & Marquis 1998].

2.3 Aspects codage et extension du formalisme

Résoudre efficacement un problème dépend à la fois du formalisme de représentation utilisé (e.g. calcul propositionnel, CSP, etc.) et du codage [19] proprement dit (e.g. les codages des problèmes de planification [Vidal 2000]). J'ai essentiellement travaillé sur les aspects liés à l'extension du formalisme.
Malgré les progrès récents obtenus sur la résolution du problème SAT et une utilisation de plus en plus fréquente d'une représentation CNF dans le cadre de la modélisation des problèmes réels (cf. introduction 1), l'utilisation de formalismes plus puissants et plus généraux reste encore une voie de recherche prometteuse. Ces travaux sont essentiellement motivés par des applications (cf. section 2.3.2) qui permettent une mise en évidence des limites des cadres existants. La tendance dans le cadre SAT, c'est d'aller vers des formalismes de manipulation de fonctions booléenes plus générales [Franco *et al.* 1997, Gu 1997, Groote & Warners 2000, Sheeran & Stalmarck 2000]. Dans le cadre des CSP, on commence à passer du cadre des CSP binaires au CSP n-aires [Régin & eds 1998, Bessière 1999, Smith *et al.* 2000].

2.3.1 Logique à cardinalité

Les travaux réalisés (cf. 2.2) autour de l'exploitation des propriétés structurelles nous ont conduits a rechercher un nouveau formalisme pour exprimer de manière naturelle certains problèmes et plus particulièrement les propriétés structurelles comme la symétrie.

L'approche qui consiste à rechercher une meilleure façon d'exprimer certaines contraintes particulières était très utilisée en programmation logique par contraintes (CLP). On peut citer, les contraintes symboliques (non numériques), par exemple sur des ensembles et leurs éléments (e.g. contraintes Tous Différents « *allDiff* », opérateur de cardinalité [Hentenryck & Deville 1990]). Les variables utilisées dans ce type de contraintes sont en général symétriques. L'utilisation de telles contraintes en CLP a souvent été motivée par des applications réelles (comme les problèmes de planification, d'ordonnancement, etc.) ou encore par le besoin d'exprimer des contraintes disjonctives.

L'utilisation d'opérateurs de cardinalité du type de celui qu'a introduit Pascal Van Hentenryck en programmation logique avec contraintes (CLP) [Hentenryck & Deville 1990] permet une généralisation naturelle du formalisme clausal. La connaissance est donc décrite par un ensemble de couples $(\alpha, \mathcal{L})$ où $\mathcal{L}$ est une liste de littéraux et α le nombre minimal de littéraux de $\mathcal{L}$ qui doivent être satisfaits dans tout modèle de $\mathcal{L}$. Ce formalisme permet l'expression des deux variantes suivantes : la formule $(\alpha, +, \mathcal{L})$: "au plus α littéraux de $\mathcal{L}$ doivent être interprétés à $vrai$", et la formule $(\alpha, e, \mathcal{L})$: "exactement α littéraux de $\mathcal{L}$ doivent être interprétés à $vrai$". Cette nouvelle représentation induit une meilleure expression (plus compacte et naturelle) des problèmes et de leurs propriétés de symétrie [20]; en outre, elle accroît la puissance de discri-

18. Pour chaque littéral l on calcule la liste des littéraux impliqués par propagation unitaire par $\mathcal{F} \wedge l$.
19. Voir compte rendu du sous-groupe de travail du GT12 "Influence du codage en calcul propositionnel" : http ://www.lim.univ-mrs.fr/ gtalgo.
20. Une formule de cardinalité est totalement symétrique.

mination des conditions préalables à la détection des symétries. Par conséquent, la détection des symétries y est moins coûteuse.

Nous avons associé à ce formalisme le développement d'une méthode de résolution (complète et décidable). A l'aide de cette dernière, nous avons donné une nouvelle preuve de complexité linéaire pour le problème des pigeons (sans utilisation des propriétés de symétrie) **[Benhamou *et al.* 1994]**.

De plus, une propriété d'équivalence entre la représentation clausale classique et celle des formules de cardinalité nous a permis d'adapter efficacement certains outils de la résolution classique (méthodes et raffinements) **[Saïs 1993]**.

L'efficacité de la procédure de Davis et Putnam, adaptée au cadre d'une représentation utilisant les formules de cardinalité, est sensiblement améliorée. En outre, l'introduction des symétries, dont les symétries fortes, a permis une amélioration supplémentaire. En effet, une preuve de complexité linéaire est donnée pour le calcul du nombre de Ramsey $R(3,3,3)$ (cf. **[Benhamou *et al.* 1992a, Benhamou & Saïs 1994]**).

On peut trouver dans [Graham *et al.* 1980] une description des nombres de Ramsey trouvés et encore recherchés. Les résultats que nous avons obtenus sur ce sujet grâce à une utilisation combinée des formules de cardinalité et des symétries ouvrent des perspectives quant à la recherche de tels nombres. Le problème qui se pose est certainement celui de trouver un codage plus efficace.

Dans le cadre du projet BAHIA[21] dont l'objectif était d'analyser et de comparer différentes approches (l'approche logique, CSP et programmation linéaire 0/1 (PL 0/1)), le formalisme à cardinalité a constitué une brique importante dans l'étude des pontages entre la programmation linéaire 0/1 et la logique propositionnel (voir **[Bahia 1995]** pour les pontages réalisés entre les trois formalismes). De nombreux travaux ont été réalisés par Hooker autour de l'utilisation des techniques d'optimisation combinatoire en logique (e.g. [Chandru & Hooker 1999]) et dans le cadre CSP (e.g. [Hooker 2000]). Le même auteur a proposé récemment une généralisation des formules de cardinalité, en une représentation sous forme de règles dont la partie gauche et droite sont des formules de cardinalité[22][Hooker & Yan 1999]. D'autres travaux ont été également réalisés autour de l'extension des algorithmes de SAT à la résolution de problèmes d'optimisation combinatoire [Walser 1997, Walser 1999] (pour la recherche locale) et [Barth 1995] (pour la procédure DPL).

2.3.2 Formalisme étendu

Une collaboration est née d'une discussion avec Antoine Rauzy sur l'intérêt de disposer d'un formalisme booléen plus expressif, en vue de modéliser et de résoudre des questions posées dans le cadre d'applications industrielles, en particulier les problèmes liés à l'analyse de sûreté de fonctionnement de systèmes industriels. Les problèmes étant de taille considérable, la représentation CNF s'est avérée très limitée.

Notre objectif est d'étendre au maximum le formalisme booléen tout en :

- restant dans le cadre du calcul propositionnel, lequel nous semble adapté à la résolution de nombreux problèmes pratiques ;
- gardant les mêmes structures de données, en tout cas les mêmes principes de mise en œuvre des algorithmes (algorithmes énumératifs, techniques de recherche locale et les diagrammes binaires de décision) ;

21. BAHIA : Booléen, Algorithmes et Heuristiques pour l'IA.
22. une règle de cardinalité de la forme suivante : $(k, x_1, \ldots, x_m) \Rightarrow (l, y_1, \ldots, y_n)$.

– gardant la même complexité algorithmique sur les formules étendues que sur les formules CNF et surtout la même efficacité pratique.

Nous avons montré que cela est possible sur des circuits combinatoires à plusieurs niveaux et construits avec des fonctions de seuil. Nous avons défini un nouveau formalisme généralisant les formules de cardinalité introduites dans ma thèse. Ce dernier étend les ensembles de clauses dans deux directions :

1. Des formules à deux niveaux (conjonctions de disjonctions) aux formules à un nombre arbitraire de niveaux.
2. Des clauses aux fonctions de seuil.

Plus précisément, une contrainte est représentée par l'équation (ou porte) suivante :

$$s \;=\; \#(\alpha_1.e_1, \ldots, \alpha_k.e_k) \in [\beta_{min}, \beta_{max}](P) \tag{2.1}$$

où s (sortie) et $\{e_1, \ldots, e_k\}$ (entrées) sont des variables propositionnelles, et $\{\alpha_1, \ldots, \alpha_k\}$ (poids) et $\{\beta_{min}, \beta_{max}\}$ (seuils) sont des entiers positifs ou nuls. P est $vrai$ si et seulement si la sortie s et la double inégalitée $\beta_{min} \leq \sum_{i=1,..,k} \alpha_i.e_i \leq \beta_{max}$ sont vérifiées ou infirmées simultanément. Une formule (circuit) est une conjonction d'équations (portes). Elle peut être vu comme un (hyper)graphe orienté acyclique dont les sommets sont les variables et dont les (hyper)arcs relient les entrées de chaque porte avec sa sortie.

Signalons quelques avantages de l'extension proposée sur les formules CNF :

– elle est beaucoup plus concise. Par exemple, le fait que k formules parmi n doivent être vraies se code en une seule équation ;
– elle permet facilement l'introduction de lemmes, c'est-à-dire de variables codant des formules ;
– elle permet de tirer partie facilement des symétries ou de toutes autres régularités apparaissant dans les problèmes, en les détectant automatiquement comme dans **[Benhamou & Saïs 1992]** ou en écrivant directement les formules de façon à en éliminer les redondances [23] ;
– elle permet de mettre en œuvre des procédures de réécriture des formules dans l'esprit du vérificateur de modèle INSTEP [Vlach 1993]. INSTEP implémente deux types de règles : les filtrages (ou simplifications immédiates) des formules et la décomposition de Shannon qui consiste à remplacer une formule $\mathcal{F}$ par la formule $(x \wedge \mathcal{F}_{x\leftarrow 1}) \vee (\neg x \wedge \mathcal{F}_{x\leftarrow 0})$.

Grâce à des structures de données adaptées, de nombreux algorithmes classiques (filtrages, énumération, recherche locale, diagramme binaires de décisions) ont été étendus sans difficultés particulières et surtout sans augmentation excessive de la complexité **[Rauzy *et al.* 1999]**.

Ce travail est un premier pas vers la mise en place d'un formalisme booléen plus général. Il reste beaucoup de travail à faire sur ce sujet, en particuliers sur :

– l'étude de techniques de simplification (pré-traitements) ;
– la mise en place d'heuristiques qui tiennent compte de la structure particulière du formalisme ;
– l'étude sur les aspects de compilation de formules booléennes (recherche de couverture d'impliquants (impliqués) premiers) ;
– l'extension des algorithmes pour résoudre des problèmes d'optimisation (ce que permettent les poids associés aux variables dans les équations).

23. Par exemple en imposant, dans un problème de Ramsey, des inéquations sur le nombre d'arêtes de chaque couleur.

2.4 Aspects algorithmiques

2.4.1 Recherche locale

Depuis le début des années 90, de nombreuses techniques à base de recherche locale ont été développées pour résoudre SAT. Mis à part le principe de base de l'algorithme, les stratégies utilisées pour échapper des minima locaux (e.g. « *break-out* » de MORIS[24] [Morris 1993], random walk[25] de SELMAN [Selman & Kautz 1993]) sont très différentes de celles utilisées en optimisation combinatoire. Cette constatation simple nous a conduits à étudier la possibilité d'adapter au cadre SAT des techniques de recherche locale développées en recherche opérationnelle (e.g. le recuit simulé, la méthode Tabou) et de les comparer aux autres approches à la GSAT. Par sa simplicité de mise en œuvre, notre choix s'est porté sur la méthode Tabou [Glover 1989]. La première version simple de la méthode Tabou que nous avons développée pour SAT (appelée TSAT) utilise une liste d'interdits de longueur fixe sans critère supplémentaire (e.g. critère d'aspiration) et s'est révélée plus efficace que les meilleures méthodes développées à ce moment là (GSAT random-Walk). La clé de l'efficacité de cette méthode réside dans le réglage de ses paramètres, c'est-à-dire la taille de la liste Tabou. Cette étude a donc permis de confirmer l'extrême sensibilité des techniques de recherche locale au réglage de leurs paramètres. Un autre résultat intéressant obtenu sur des instances aléatoires $k - SAT$, méritant une étude analytique approfondie, est celui montrant l'évolution linéaire de la taille optimale (obtenue empiriquement) de la liste tabou en fonction du nombre de variables. Nous avons aussi étudié l'influence de l'interprétation initiale sur l'efficacité de TSAT. Pour ce faire, l'interprétation initiale est générée en exploitant le déséquilibre entre les occurrences des variables. Des améliorations ont été obtenues sur les instances structurées. En revanche cette modification donne des résultats comparables à TSAT sur les instances aléatoires. Signalons que TSAT enlève le caractère aléatoire introduit dans GSAT, ce qui simplifie considérablement l'algorithme **[Mazure *et al.* 1997a]**.

L'efficacité de la méthode de recherche locale Tabou dépend considérablement de la taille de la liste d'interdits. Son réglage est difficile à obtenir particulièrement sur les instances structurées. Une voie que nous explorons actuellement concerne le réglage dynamique de ce paramètre crucial. Nous avons établi une relation entre la taille des cycles rencontrés au cours de la recherche et la longueur de la liste tabou. Des résultats prometteurs ont déjà été obtenus sur les instances aléatoires $k-SAT$. Notre approche exploite les informations liées à la présence et à la taille des cycles rencontrés au cours de la recherche. En effet, sur les instances $k - SAT$, la taille de la liste tabou converge vers la taille optimale obtenue précédemment. Malheureusement, ces informations restent insuffisantes pour établir le réglage optimal sur certains problèmes structurés.

2.4.2 Recherche systématique

2.4.2.1 Généralisation du théorème de partition du modèle

Comparativement aux améliorations obtenues par l'utilisation de filtrages plus puissants, d'heuristiques judicieuses, la recherche de propriétés permettant d'élaguer l'arbre de recherche a reçu moins d'attention.

Une des techniques les plus connue au moins dans la communauté française est sans doute le théorème de partition des modèles [Jeannicot *et al.* 1988, Oxusoff & Rauzy 1989]. Le théorème peut s'expri-

24. La méthode appelé Break-Out effectue une pondération des clauses. Cette pondération « poids » d'une clause est utilisée ensuite dans la fonction d'évaluation. Le poids d'une clause représente le nombre de fois qu'elle a été falsifiée depuis le début de la recherche.

25. Dans la stratégie Random-Walk, la prochaine variable à inverser est choisie parmi celles apparaissant dans les clauses falsifiées. Avec une probabilité p, celle améliorant le plus la fonction objective, et avec une probabilité $1 - p$ la variable est choisie au hasard.

mer comme suit : Si $\mathcal{F}_k \subset \mathcal{F}_j$, alors $\mathcal{F}_j$ est satisfiable ssi $\mathcal{F}_k$ est satisfiable ($\mathcal{F}_k$ et $\mathcal{F}_j$ étant des ensembles de clauses). Cette propriété a été intégrée dans une procédure DPL. Cette variante appelée « Evaluation Sémantique » a permis une amélioration sensible sur certaines classes de problèmes structurés.

Une généralisation immédiate consiste à remplacer l'inclusion syntaxique par la réciproque de la relation de déduction logique. Pour des raisons d'efficacité évidente, nous avons utilisé une relation plus faible (la déduction par propagation unitaire). Nous avons proposé deux façon d'intégrer cette notion dans la procédure DPL. Une première utilisation consiste à l'appliquer au moment du retour-arrière, la seconde avant le choix de la variable de branchement. Dans les deux cas, on peut intégrer la propriété de coupure dans un algorithme utilisant, soit les heuristiques traditionnelles, soit des heuristiques (du type de celles mises en place par OXUSOFF et RAUZY) spécifiques favorisant l'application du théorème. Une étude assez avancée autour de ces questions est actuellement en cours. Une autre généralisation consiste à associer (au cours de la recherche) pour chaque (sous-)clause de la base le niveau (ou profondeur de l'arbre) à partir duquel elle est apparue ou "touchée". Cet étiquetage des clauses de la formule au cours de la recherche permet de savoir à tout moment le niveau de l'arbre où une sous-clause a été touchée (apparue). Cette information est ensuite prise en compte lors de l'application du théorème. Plus précisément, lors de la production d'une sous-clause par propagation unitaire, nous avons montré qu'en analysant le graphe d'implications, on peut déterminer le niveau de l'arbre à partir duquel la sous-clause est produite. Ce qui nous permet de réaliser un retour-arrière non chronologique et de produire de nouvelles sous-clauses, importantes pour la suite de la recherche.

Ce travail a été développé en collaboration avec Éric GRÉGOIRE, Pierre MARQUIS et Bertrand MAZURE. Une description plus détaillée des différents points développées ci-dessus peut être trouvée dans **[Mazure 1999]**. La partie expérimentale est actuellement en plein développement (cf. chapitre 4).

2.4.2.2 Poids : apprentissage au cours de la recherche

De nombreuses techniques ont été proposées pour améliorer les algorithmes énumératifs en essayant d'analyser les causes d'échec. Elles font appel souvent à l'ajout de nouvelles clauses, ou à des techniques sophistiquées d'analyse de conflits, qui induisent souvent un sur-coût qui peut être non négligeable. En outre, la plupart des heuristiques traditionnellement développées utilisent des fonctions de sélection basées sur des paramètres syntaxiques (longueur des clauses, nombre d'occurrences, etc.). Par exemple, le score d'une variable peut être calculé comme suit : $F(x) = f(x) + f(\neg x)$, avec $f(x) = \sum_{x \in C} w(C)$, et $w(C) = 2^{|C|}$. De plus, la même heuristique est ensuite utilisée tout le long de la recherche, bien que les sous-problèmes rencontrés (formule simplifiée par l'interprétation partielle courante) au cours de la recherche n'ont pas forcément les mêmes caractéristiques que le problème initial.

Notre objectif, qui sans nul doute peut paraître ambitieux, concerne la mise au point d'une heuristique adaptative évoluant au cours de la recherche, en mettant à profit ces échecs répétés.

Ci-dessous quelques arguments expliquant les motivations de cette étude. On sait que prouver l'inconsistance d'une formule est souvent plus difficile que de prouver la consistance. De nombreuses instances inconsistantes difficiles contiennent une (des) sous-formules inconsistante(s). Une heuristique n'étant pas à l'abri d'une erreur dans le choix des variables de branchement, elle peut conduire à la résolution répétée du même sous-problème inconsistant (cf. section 2.4.3).

En d'autres termes, notre objectif est de mettre au point une heuristique permettant d'éviter de rencontrer les mêmes situations d'échec. L'heuristique que nous avons mise au point utilise un facteur de pondération supplémentaire pour les clauses, en comptabilisant au cours de la recherche le nombre de fois que chaque clause est à l'origine de la contradiction (ce qui n'est pas très coûteux en pratique). Plus précisément, on obtient pour la fonction $F(x)$ décrite précédemment, une nouvelle fonction $F'(x) = f(x) + f(\neg x)$ avec $f(x) = \sum_{x \in C} \alpha(c) \times w(C)$ et $\alpha(C)$ le facteur supplémentaire ajouté, qui représente le nombre de fois que la clause C a été la cause de la contradiction.

L'augmentation du poids d'une clause revient à rendre cette clause redondante et de plus en plus importante. Cette information est directement prise en considération par la fonction de choix.

Les expérimentations réalisées sur la base d'instances (DIMACS) montrent clairement son efficacité par rapport à une version de base **[Brisoux *et al.* 1998, Brisoux *et al.* 1999]**. Sur certaines classes de problèmes, les résultats obtenus sont meilleurs que ceux des meilleurs algorithmes connus.

Poursuivant le même objectif, Bruni et Sassano[Bruni & Sassano 2000] ont présenté un papier au dernier Workshop SAT 2000, qui utilise une idée similaire.

2.4.3 Méthode mixte : coopération entre approche systématique et non systématique

La recherche locale ne peut prouver l'inconsistance. Sur une instance consistante de grande taille et non élémentaire, à cause de sa rigidité, une recherche systématique a peu de chance de trouver un modèle. Ces différentes techniques montrent donc une certaine complémentarité qui peut être illustrée par le problème suivant :
énoncé du problème : J'ai perdu une clé dans la maison et j'ai besoin de la trouver rapidement. Quel type de méthode devrais-je utiliser ?
solution possible : On peut procéder à une fouille systématique de toute la maison en commençant par exemple par le garage. Cependant, l'expérience montre que ce type de recherche est le dernier recours, une recherche non systématique consistant à suivre son instinct permet souvent de trouver plus rapidement l'objet recherché.

Il est donc naturel de penser à une approche permettant de faire coopérer fortement les deux types de techniques. La question est : comment les faire coopérer ?

La solution du problème précédent montre qu'il est judicieux de commencer par une recherche locale (incomplète) avant de décider de lancer une recherche systématique. Dan un premier schéma, la recherche locale est utilisée comme pré-traitement. Ce premier schéma pose la question suivante : est-il possible de bénéficier du travail effectué lors de la première phase, afin d'accélérer la seconde phase ? Une réponse possible consiste à éviter lors de la seconde phase de visiter autant que possible les endroits où la recherche a été infructueuse lors de la première phase.

Un second schéma possible consiste à combiner plus fortement les deux types d'algorithmes : appliquer localement le premier schéma au niveau de chaque pièce visitée. Dans cette seconde approche, si l'ensemble des pièces de la maison ont été visitées, on obtient une méthode complète.

Préalablement à la mise en œuvre de tels schémas de coopération dans le cadre de la résolution du problème SAT, il faut trouver des informations pertinentes à capitaliser lors du processus de recherche locale, pour fournir ensuite des indications susceptibles d'améliorer la phase de recherche systématique.

On peut être tenté de répondre que les informations en question sont en fait les interprétations visitées lors de la recherche locale. Il est bien évident que fournir un tel ensemble n'est pas réaliste. Cependant, on sait qu'une clause C est falsifiée par $2^{n-|C|}$ interprétations possibles, on peut par conséquent relever que plus une clause est falsifiée au cours de la recherche locale, plus l'espace des interprétations falsifiant une telle clause tend à être exploré. Cette information intéressante est plus facile à exploiter. En effet, on peut au cours de la recherche systématique essayer de satisfaire de telles clauses et ainsi éviter (partiellement) les zones déjà explorées par l'algorithme de recherche locale.

Un des objectifs de cette étude est la mise au point de techniques permettant de prouver efficacement différentes formes d'inconsistance, un problème central en représentation des connaissances et en démonstration automatique. En effet, l'inconsistance d'une base de connaissances est très fréquemment due à la présence accidentelle d'une information contradictoire sur un sujet donné. Cette inconsistance pollue globalement la base de connaissances (i.e. toute information ainsi que son contraire peuvent être inférés de cette base). Malheureusement, il n'existe pas de méthode universelle et efficace pour détecter et/ou localiser ces inconsistances.

Une technique de recherche locale (incomplète) ne peut prouver l'inconsistance. Cependant, elle peut fournir des indications pertinentes sur de telles inconsistances locales. En effet, en comptabilisant le nombre de fois que chaque clause est falsifiée au cours de la phase de recherche locale, on s'est aperçu expérimentalement (sur de nombreux problèmes de DIMACS [Dimacs 1993]) que les clauses les plus souvent falsifiées sont les plus susceptibles d'appartenir aux sous-ensembles minimalement inconsistants. Ce qui nous a permis de mettre en évidence une technique capable de localiser l'inconsistance. Le nombre de fois qu'une clause est falsifiée (« trace ») lors de la recherche locale constitue une information pertinente que nous avons utilisée dans le cadre des deux schémas de coopération précédents. Dans le premier schéma, cette trace est exploitée pour trouver un bon ordonnancement des variables pour la méthode complète. Dans le second schéma, nous avons utilisé dans la procédure DPL la recherche locale comme un oracle, qui permet non seulement d'étendre l'interprétation partielle courante, mais également de fournir en cas d'échec la prochaine variable à affecter (celle apparaissant le plus souvent dans les clauses falsifiées). Notons que les deux schémas proposés sont complets. Des améliorations surprenantes ont été obtenues sur de nombreux problèmes de la base de test de DIMACS **[Mazure 1995, Mazure *et al.* 1996, Mazure *et al.* 1998]**. D'autres schémas de coopération entre les algorithmes de recherche locale et les algorithmes complets de recherche systématique sont présentés dans **[Mazure 1999]**.

Il est évident que les problèmes les plus difficiles présentent une inconsistance globale (l'inconsistance est due à l'ensemble du problème au lieu d'un sous-ensemble de celui-ci) ; ces derniers sont généralement construits de manière artificielle ou aléatoire et ne se retrouvent pas dans les applications réelles. De plus, ce genre d'instances possèdent le plus souvent de très fortes régularités (problèmes des pigeons, etc.). Une étude comparative avec l'approche utilisant la résolution bornée comme prétraitement a été réalisée **[Mazure *et al.* 1997a]**. Ces travaux ont été réalisés en collaboration avec Éric GRÉGOIRE et Bertrand MAZURE.

D'autres types d'approches complètes utilisant la recherche locale ont été développées. On peut citer dans ce cadre l'algorithme SCORE [Chabrier *et al.* 1991], et SCORE(FD/B) [Chabrier *et al.* 1995]. D'autres algorithmes complets à base de recherche locale ont été obtenus par ajout de clauses [Ginsberg 1993, Hao & Tétard 1996]. Des approches similaires au premier schéma de coopération décrit précédemment ont été étudiées indépendamment par Thierry CASTELL [Castell 1997] et James M. CRAWFORD [Crawford 1993]. Un inventaire complet des méthodes mixtes dans le cadre des problèmes de satisfaction de contraintes est donné dans [Lobjois & Lemaître 1997].

2.5 Phénomène de seuil et génération aléatoire

Dès lors que l'intérêt de la génération aléatoire d'instances SAT est avéré, d'une part par le résultat de Chvàtal et Szemerédi montrant qu'elles sont difficiles pour la résolution et donc pour toute les méthodes énumératives du type procédure de Davis et Putnam, d'autre part par la mise en évidence de phénomènes de seuil, plusieurs questions se posent :

1. Est-il possible de mettre en évidence l'existence de conditions nécessaires et/ou suffisantes à la caractérisation des ensembles de clauses encore plus difficiles ? En utilisant, par exemple, des conditions nécessaires à la présence de symétries ?
2. Les instances générées par le modèle standard contiennent environ 50% de clauses de Horn. Que se passe-t-il, si l'on fait varier la probabilité de négativer un littéral de 0.5 à 1 ?
3. Est-il possible de générer aléatoirement des instances difficiles toujours satisfiable ?
4. Est-il possible d'établir des modèles de génération aléatoire donnant lieu à des instances plus proches des problèmes réels ?

Avec Richard GÉNISSON, nous nous sommes particulièrement intéressés aux deux premières questions et plus accessoirement à la troisième.

Mes travaux de recherche précédents sur les symétries ont conforté ma conviction que les problèmes les plus difficiles présentent une structure très symétrique. Ce qui nous a conduits à répondre à la première question, en étudiant deux nouveaux modèles de génération d'ensembles de clauses régulières (toutes les variables apparaissent le même nombre de fois, les clauses ayant toutes la même longueur). Dans le premier modèle, tous les littéraux ont exactement le même nombre d'occurrences ; dans le second, les variables apparaissent le même nombre de fois, avec un déséquilibre entre le nombre d'occurrences des littéraux positifs et négatifs. Dans ces deux modèles, des conditions nécessaires pour l'existence de symétrie sont vérifiées. Pour ces deux modèles, nous avons montré expérimentalement que le seuil se situe pour 3-SAT au rapport $n/m = 3.6$ (chaque variables apparaît 6 fois positivement et 5 fois négativement). Les instances obtenues sont beaucoup plus difficiles que pour le modèle standard. Les instances satisfiables posent plus de problèmes aux algorithmes de recherche locale. D'autre part, l'étude que j'ai réalisée sur le modèle de génération standard en variant la probabilité de tirer un littéral négatif, nous a conduits à établir des liens entre la valeur du seuil et la distribution des occurrences des variables (déséquilibre entre littéraux positifs et négatifs, proportion de clauses de Horn). Ces différents résultats sont publiés dans **[Génisson & Saïs 1994, Mazure 1999]**.

D'autres auteurs ont étudié ensuite le modèle de génération d'ensembles de clauses régulières [Bayardo & Schrag 1996, Dubois & Boufkhad 1996]. On peut également trouver des générateurs aléatoires d'instances difficiles pour la recherche locale [Castell & Cayrol 1997, Rauzy 1995a, Achlioptas *et al.* 2000] et des tentatives de prise en compte de certaines caractéristiques de problèmes réels dans la génération aléatoire [Gent & Walsh 1994].

Chapitre 3

Problèmes autour de SAT

Sommaire

3.1 Problèmes de satisfaction de contraintes

Les logiciels commerciaux tels que « *Ilog Solver* » et les outils pour la recherche et l'enseignement comme « *CHOCO* » montrent la maturité des recherches conduites dans la cadre de la résolution des problèmes de satisfaction de contraintes. Le formalisme et le raisonnement par contraintes offrent un outil de modélisation et de résolution majeur pour de nombreuses applications aussi diverses que le traitement du langage naturel, la gestion et l'allocation de ressources, les emplois du temps, la vision, etc. Un réseau de contrainte est défini par un ensemble de n variables $X = \{x_1, x_2, \ldots, x_n\}$ prenant leurs valeurs dans des domaines finis et discrets $D = \{d_1, d_2, \ldots, d_n\}$. Le problème de satisfaction de contraintes consiste à décider s'il existe une affectation de valeurs aux variables de manière à satisfaire un ensemble de contraintes $C = \{c_1, c_2, \ldots, c_m\}$. Une contrainte c_i est définie sur un sous-ensemble de variables $c_i = \{x_{i1}, x_{i2}, \ldots, x_{in_i}\}$ de X, où n_i est l'arité de c_i. Une telle contrainte peut être spécifiée (en extension) par une relation $r_i = \prod_{x_i \in c_i} d_i$ des combinaisons de valeurs (ou tuples) autorisées pour les variables de c_i. Ces contraintes peuvent aussi être spécifiées en intension par une fonction caractéristique (prédicat, fonction, etc). Satisfaire une contrainte c_i revient à trouver les valeurs à affecter pour chaque variable de c_i de manière à satisfaire la relation r_i associée à la contrainte.

Le problème de satisfaction de contraintes (CSP) est donc une généralisation du problème SAT. En effet, toute instance de SAT peut être vue comme une instance de CSP dans laquelle toutes les variables ont pour domaine $\{0, 1\}$, et il y a autant de contraintes que de clauses. Toute instance de CSP peut également être transformée en temps polynômiale en une instances de SAT équivalente pour l'existence de solution (les deux problèmes sont NP-complets). Les CSP binaires (où toute les contraintes ont une arité égale à 2) ont été les plus étudiés. Leur représentation en terme de graphes de contraintes (au lieu d'hypergraphes de contraintes pour les CSP n-aires) simplifie considérablement les algorithmes.

La description ci-dessus donne un aperçu des liens étroits entre SAT et CSP. Ce qui explique certaines similitudes entre les deux problèmes à la fois dans les questions posées, mais également dans les approches utilisées pour y répondre.

En effet, on retrouve par exemple de nombreux travaux traitant de l'amélioration de la procédure de résolution classique appelée généralement « *Backtrack* » (BT). Pour l'essentiel, ces différentes améliorations ont concerné les techniques de filtrage, les heuristiques de choix de variables et de valeurs et le traitement des échecs. Pour ce qui est des aspects algorithmiques, le meilleur algorithme de résolution de CSP (MAC, pour « *Maintaining Arc Consistency* ») est obtenu en maintenant dynamiquement dans « *Forward Checking* » (FC), un CSP arc consistant [1] à chaque nœud de l'arbre de recherche [Sabin & Freuder 1994]. L'heuristique proposée (minimum domaine/degré [2]) par Régin et Bessière [Bessière & Régin 1996] améliore considérablement l'algorithme. D'autres approches sont basées sur la décomposition du CSP. Ces techniques exploitent les propriétés structurelles des CSP [Dechter & Pearl 1989, Dechter 1990, Jégou 1993]. En outre, les propriétés structurelles du graphe de contraintes (e.g. largeur du graphe) et la sémantique des contraintes (contraintes fonctionnelles, etc.) sont à la base de la plupart des classes polynômiales exhibées dans le cadre CSP. Par exemple, la consistance d'arc (AC) suffit pour résoudre des CSP structurés en arbre, alors que les CSP 0/1/all [Cooper *et al.* 1994] peuvent être résolus polynômialement en utilisant la consistance de chemin (PC).

De nombreux travaux ont concerné l'étude des liens entre SAT et CSP. Dans le cadre du projet BAHIA **[Bahia 1995, Bahia 1992]**, on peut citer par exemple, le résultat de GÉNISSON et JÉGOU montrant l'équivalence entre la procédure de Davis et Putnam et l'algorithme de « *Forward Cheking* » (FC) [Génisson & Jégou 1996], les pontages entre les classes polynômiales [Génisson & Rauzy 1996], l'équivalence entre les littéraux purs et la substituabilité de voisinage [Bellicha *et al.* 1994] et par conséquent l'extension du théorème de partition du modèle au CSP ou encore le formalisme des CSP clausaux proposé dans [Castell & Fargier 1998], etc.

Avec Assef CHMEISS **[Chmeiss & Saïs 2000]**, nous nous sommes intéressés à l'étude des possibilités d'intégration forte des traitements locaux dans le cadre de la résolution de CSP. Ce travail a été essentiellement motivé par les améliorations algorithmiques obtenues dans le cadre SAT en utilisant d'une part des traitements locaux de manière plus agressive comme la propagation unitaire, et d'autre part par l'utilisation de techniques de simplification plus puissantes comme la résolution bornée. La puissance des machines fait qu'on n'hésite même plus à revenir aux vielles méthodes comme la résolution et la procédure de Davis et Putnam originale, longtemps écartée pour des raisons de complexité en espace (cf. chapitre 2).

Ce dernier point nous conduit à faire le parallèle avec la consistance de chemin qui peut être vue comme un processus de production et de vérification de contraintes. Par ailleurs, pour les mêmes raisons que pour la résolution, l'intégration de la consistance de chemin dans les algorithmes de résolution de CSP a été jusqu'à présent évitée. En effet, la complexité élevée du test de consistance de chemin (comparativement à la consistance d'arc) constitue un obstacle à sa mise en œuvre en pratique. Pour pallier cet inconvénient, nous avons proposé une exploitation partielle de cette forme de filtrage au cours de la résolution. La consistance de chemin (directionnelle) est utilisée au cours de la recherche pour effectuer des disconnections de certaines variables particulières : les variables « doubletons » (variables de degré 2). Nous avons intégré ce filtrage dans MAC en exploitant également les singletons degrés et les singletons valeurs. L'algorithme obtenu est appelé M(AC+).

1. Un CSP est k-consistent si et seulement si toute instanciation partielle de $k-1$ variables peut être étendue de manière consistante à toute $k^{ième}$ variable. La consistance d'arc (resp. la consistance de chemin) correspond à la 2-consistance (resp. 3-consistance).

2. Min dom/deg : consiste à choisir la variable maximisant le rapport entre la taille du domaine et le degré de la variable. Le degré d'une variable est égale au nombre de contraintes où elle apparaît.

Une autre technique de filtrage plus puissante que la consistance d'arc et qui nous semble particulièrement prometteuse pour aller plus loin dans la résolution de problèmes difficiles est la "singleton arc consistance" (SAC) proposée par DEBRUYNE et BESSIÈRE [Debruyne & Bessière 1997]. Plus précisément, en plus des possibilités de filtrage par SAC, nous montrons qu'il est possible de déduire des informations pertinentes supplémentaires, en analysant le processus de propagation. Nous proposons également différentes façons de l'utiliser en cours de résolution, par exemple comme moyen de raffiner les heuristiques de choix de variables.

Il reste beaucoup de travail à faire sur ce sujet. Une collaboration est actuellement en cours avec Christian BESSIÈRE (CR CNRS, LIRMM Montpellier).

3.2 Compilation des bases de connaissances et représentation compacte de l'ensemble des solutions

Afin de pallier la non-calculabilité pratique (« *intractability* ») de la déduction logique en calcul propositionnel, plusieurs approches ont été proposées. Elles s'appuient sur différents principes, en particulier :

– restriction (du langage utilisé, e.g. les clauses de Horn, ou de la relation de déduction employée, e.g. la résolution bornée) ;
– approximation (de la base de connaissances ou de la relation de déduction) ;
– compilation. L'idée est de construire une structure de données σ^* (une compilation de la base initiale σ) qui se comporte comme σ pour l'interrogation mais nécessite seulement un temps polynomial (en fonction de $\mid \sigma^* \mid$).

En collaboration avec Yacine BOUFKHAD, Éric GRÉGOIRE, Pierre MARQUIS et Bertrand MAZURE, nous nous sommes particulièrement intéressés à ce dernier principe. Nous avons proposé une nouvelle approche de compilation préservant l'équivalence logique avec la base de connaissances initiale et appelée couverture tractable (une couverture tractable est une disjonction de formules tractables). Cette nouvelle approche a été formalisée dans un cadre général. Deux cas spécifiques ont été considérés :

1. les interprétations partielles sont utilisées pour « raboter » ou simplifier la base de connaissances en un ensemble de formules traitables (simplifications traitables).
2. les interprétations partielles sont utilisées de manière à dériver un ensemble de formules Horn- renommables (hyper-impliquants).

Dans le second cas, nous avons prouvé que la taille de la couverture est strictement inférieure à celle d'une couverture par impliquants premiers. Il faut noter qu'en pratique les formules tractables ne sont pas représentées, seules les interprétations partielles (obtenues par la procédure de Davis et Putnam) nous permettant de les obtenir sont réellement sauvegardées. Des résultats expérimentaux ont montré l'efficacité d'une telle approche par rapport à l'interrogation directe de la base de connaissances. Un gain exponentiel en temps et en espace est obtenu par rapport à l'approche par impliquants premiers [Castell & Cayrol 1996, Schrag 1996]. Ce travail a été publié dans **[Boufkhad *et al.* 1997]**.

Dans les approches présentées ci-dessus, le raisonnement modulo une théorie [Marquis 1995] a été utilisé lors de la construction de la base compilée dans l'objectif de réduire la taille de cette dernière. C'est cette notion, qui a rendu possible le gain en temps et en espace par rapport aux approches par impliquants premiers. Pour de nombreux problèmes et en particulier les problèmes sous-contraints, la taille de la compilation peut être prohibitive et donc difficile à construire et à représenter. Trouver une représentation plus compacte de la formule compilée présente par conséquent un intérêt pratique indéniable pour de nombreuses applications.

Dans un cadre restreint aux formules contenant des symétries, je me suis intéressé au problème du calcul d'une représentation plus compacte de l'ensemble des solutions. En effet, pour ces formules, l'ensemble des solutions peut être partitionné en classes d'équivalence. L'utilisation de ces symétries permet de représenter l'ensemble des solutions par un sous-ensemble de solutions caractéristiques (solutions non symétriques). Une étude détaillée a été publiée dans **[Saïs 1994a, Saïs 1994c]**.

3.3 Des techniques efficaces pour SAT au secours de la révision de croyances

De nombreuses approches de la révision de croyances ont été proposées cette dernière décennie. La plupart d'entre elles ont été étudiées principalement au niveau dit de la connaissance [Newell 1982]. En dépit de quelques exceptions notables (e.g. [Papini 1991, Nebel 1992, Liberatore & Schaerf 1996]), elles n'ont pas été implantées, pour des raisons évidentes de complexité dans le pire des cas. Cependant, de récents progrès dans la résolution pratique du problème SAT ouvrent des perspectives de calculs effectifs pour les approches syntaxiques de la révision de croyances (voir e.g. [del Val 1994, Makinson & Gärdenfors 1991, Moinard 1994, Zhang *et al.* 1997]). En collaboration avec Éric GRÉGOIRE, Pierre MARQUIS et Brigitte BESSANT, nous avons proposé une nouvelle approche de la révision syntaxique des croyances dans un cadre non monotone. Motivée par des considérations pratiques cette approche est basée sur une approximation de la réunion des sous-ensemble minimaux inconsistants obtenus grâce à l'application des techniques complètes à base de recherche locale. Cette approche comporte un processus de résolution d'inconsistance en deux étapes. En premier lieu, une utilisation disciplinée d'ingrédients non monotones (des propositions d'anormalité qui permettent de représenter des règles de défaut et des exceptions) est offerte à l'ingénieur de la connaissance pour éviter de nombreuses inconsistances qui apparaîtraient si une représentation et une interprétation logique standard des croyances étaient réalisées. Les inconsistances restantes sont supposées imprévisibles et révisées par affaiblissement de formules apparaissant dans toute sous-base minimalement inconsistante, comme si elles représentaient des cas exceptionnels qui en fait n'ont pas lieu. Ce dernier point nous a conduit a utiliser les mêmes ingrédients que dans la première étape, pour affaiblir et rétablir la consistance. L'affaiblissement d'une formule est obtenue par ajout disjonctif de nouvelles propositions ; ceci offre des perspectives intéressantes en ce qui concerne les révisions itérées.

Alors que le calcul de bases de connaissances révisées demeure intraitable dans le pire des cas, notre approche bénéficie d'une technique heuristique efficace de recherche locale qui est souvent viable en pratique. Dans le même état d'esprit, des techniques de recherche locale spécifiquement adaptées pour MAX-SAT sont étudiées pour implanter une politique de révision alternative qui minimise le nombre de formules à affaiblir afin de rétablir la consistance **[Bessant *et al.* 2000, Bessant *et al.* 1998]**.

3.4 Une méthode complète pour des bases de connaissances propositionnelles non monotones

Nous avons proposé une nouvelle approche pour calculer de façon effective des inférences au sein d'une logique propositionnelle non monotone simple, fondée sur un concept de modèles préférés. L'approche que nous proposons est originale à deux égards au moins. Premièrement, elle se base sur des techniques de recherche locale tout en préservant la complétude logique. En second lieu, elle est efficace d'un point de vue expérimental pour une classe importante de grandes bases de connaissances non monotones. Plus précisément, nous étendons à un cadre non monotone des résultats heuristiques récents liés à la résolution pratique du problème de satisfiabilité d'un ensemble de clauses (SAT). De fait, la procédure de preuve employée se base sur l'utilisation d'une méthode de recherche locale pour SAT et sur une heuristique efficace lorsque cette recherche locale n'a pu exhiber un modèle. Nous l'employons au sein d'un formalisme de représentation permettant des règles de raisonnement par défaut avec priorités exprimées à l'aide de propositions d'anormalités à la McCarthy. Un domaine d'applications typique concerne les formes de raisonnement révisable qui peuvent être tenues au sujet d'un modèle « profond » d'un système ou d'un appareillage complexe, où les propositions d'anormalité sont utilisées pour représenter les défaillances possibles de composants et où un nombre limité de pannes peuvent survenir simultanément **[Grégoire *et al.* 1998]**.

3.5 Coopération consistante et non-monotonie

Notre expérience dans le cadre de la détection et de la localisation des inconsistances nous a conduits à envisager l'extension et l'application de ces techniques pour résoudre les problèmes liés à la fusion et à l'interaction des bases de connaissances dans un cadre de travail coopératif. En effet, le problème de maintien de la consistance lorsque l'on fait interagir différentes sources d'informations est fondamental dans de nombreuses applications : bases de données distribuées, systèmes multi-agents, travail coopératif, etc. Dans ce cadre, nous avons montré que les techniques que nous avons développées dans le cadre du problème SAT peuvent être utilisées pour localiser les inconsistances qui peuvent apparaître lors de la fusion de bases de connaissance. Cette étude a été réalisé dans le cadre du projet inscrit au sein du contrat de plan État Région « Ganymede : Travail coopératif et communications avancés ». Une partie de cette étude est publiée dans le journal « Int. Journal of Cooperative Information Systems » **[Mazure *et al.* 1997b]**.

3.6 Premier ordre fini

Les remarquables progrès obtenus dans la résolution du problème SAT conduisent naturellement à l'extension et l'application de ces techniques dans le cadre de la logique du premier ordre. Une approche naïve consiste à appliquer les algorithmes de SAT sur une instanciation complète de la formule du premier ordre. Cette méthode peut provoquer une explosion combinatoire. En effet, le nombre d'instanciations possibles peut être extrêmement large et même infini. Dans ce cadre, on peut citer par exemple la méthode « *ModGen* » développée par S. KIM et H. ZHANG [Kim & Zhang 1994] pour résoudre des formules du premier ordre fini. Cette technique se compose de deux modules, le premier effectue une transformation de la formule du premier ordre en formule propositionnelle sous forme CNF ; le second module utilisant les techniques de SAT pour trouver un modèle de la formule. Dans le cadre du premier ordre fini restreint aux interprétations de Herbrand, nous nous sommes intéressés au problème suivant : soit BC une base de connaissances consistante et f une formule, prouver la consistance de $BC \cup f$. Nous avons proposé une technique efficace qui effectue une instanciation partielle et incrémentale de la base à partir des prédicats de f. Cela permet d'éviter dans de nombreuses situations d'instancier complètement la base. En d'autres termes l'approche que nous avons proposée se limite à l'instanciation d'un sous-ensemble de BC en relation avec f (partageant des prédicats). La résolution se fait également de manière incrémentale au fur et à mesure du processus d'instanciation. Ces travaux ont été réalisés en collaboration avec Éric GRÉGOIRE et Laure BRISOUX **[Brisoux *et al.* 1998, Brisoux *et al.* 2000]**.

Chapitre 4

Perspectives

Sommaire

Dans ce chapitre je présente mes perspectives de recherche à court et à moyen terme. Les pistes et les détails techniques donnés dans certains cas montrent l'état de maturité et le caractère à court terme de certaines de ces perspectives.

4.1 Problème SAT

Comme dans tout domaine de recherche, les résultats obtenus montrent à la fois les progrès mais également les limites des approches utilisées. Il est plus facile de continuer à améliorer et à étendre les approches traditionnellement utilisées que de trouver de nouvelles pistes originales qui permettront éventuellement d'aller plus loin et de répondre aux limites des approches traditionnelles.

Sans prétendre à la paternité exclusive de ces pistes, je présente mes perspectives de recherche sur SAT en deux axes :

– l'extension des approches traditionnelles ;
– l'étude d'autres approches.

4.1.1 Approches traditionnelles

4.1.1.1 Aspects algorithmiques

Parmi les algorithmes de recherche systématiques à la Davis et Putnam, on peut grossièrement distinguer deux catégories d'algorithmes :

- les algorithmes prospectifs : ils obtiennent leurs meilleures performances sur les problèmes aléatoires difficiles au seuil (e.g. CSAT, POSIT, Sato, Satz, CSAT, Tableau, etc.).
- les algorithmes de type rétrospectif : il faut plutôt les utiliser pour résoudre des problèmes réels ou structurés (e.g. Relsat, Grasp, etc.).

Raisonner : Pour résoudre des problèmes réels et des applications, il me paraît primordial de s'orienter vers le développement de méthodes intégrant des formes de raisonnement (e.g. utilisation de lemmes, de règles de déduction). L'analyse des échecs est une forme de raisonnement. En effet, à l'opposé des problèmes aléatoires, les instances codant des applications réelles présentent des structures particulières sur lesquelles il faut raisonner. En d'autres termes, éviter dans ces cas là d'utiliser des algorithmes dits de « force brute ». Je pense qu'à l'avenir on trouvera plus d'algorithmes appartenant à la seconde catégorie ou encore des techniques intégrant avantageusement les deux principes.

Spécialiser : On sait que chaque algorithme admet ses formules privilégiées, et ses formules difficiles. Il semble impossible de mettre au point un algorithme qui obtiendrait les meilleures performances sur tous les problèmes. Il me paraît plus raisonnable d'essayer de développer un algorithme qui résout efficacement une classe de formules. Une technique qui résout efficacement les problèmes de circuits par exemple, mais qui peut présenter un comportement catastrophique sur d'autres types de formules peut rester pour moi un bon algorithme. Les compétitions organisées dans le cadre SAT (DIMACS et Beijin) ont largement contribué aux développement des recherches. En plus du comportement général d'un algorithme, il me paraît nécessaire de prendre en considération le comportement spécifique d'un algorithme (e.g. quelle est la meilleure méthode disponible pour résoudre des problèmes de planification ?).

Après la découverte du phénomène de seuil, Olivier DUBOIS proposait l'idée de spécialiser les algorithmes soit dans la recherche d'une solution, soit dans la recherche d'une contradiction. En effet, un algorithme peut avoir des comportements très différents selon que l'on suppose qu'il existe ou non une solution. Cette idée intéressante ne me paraît que très partiellement explorée et nécessite de l'être systématiquement. Le challenge 1. proposé dans [Selman *et al.* 1997] concerne la résolution d'instances aléatoires insatisfiables de 700 variables au seuil par une méthode complète. Cette question est très difficile (voir “autres approches”). En revanche, résoudre des problèmes satisfiables de 700 variables au seuil par des techniques complètes (spécialisées dans la recherche de solutions) me paraît dans une certaine mesure plus facile (SAT est NP-complet et UNSAT est Co-NP-Complet). Elle présente plusieurs intérêts : elle permettra de se rapprocher plus des performances de la recherche locale et d'avoir indirectement des intuitions plus précises sur l'efficacité remarquable de la recherche locale sur de telles instances, etc. De plus, cette question est loin d'être évidente. Des travaux sont actuellement en cours dans cette direction.

Combiner : La mise au point de méthodes hybrides (en combinant différents algorithmes) est un thème de recherche actuel en optimisation combinatoire et en programmation logique par contraintes.

Nous avons montré dans le cadre de nos travaux précédents qu'il est possible de combiner avantageusement différentes approches (recherche locale et recherche systématique). Dans ce cadre, nous envisageons de continuer à explorer cette voie qui s'est déjà révélée viable. En particulier, proposer d'autres schémas de combinaison ou encore exploiter d'autre types d'informations. Par exemple, dans de nombreuses situations la recherche locale peut trouver une interprétation qui satisfait toute les clauses sauf une (« Critical Truth Assignment » ; notion caractérisant de nombreux problèmes difficiles). Cette clause particulière non satisfaite est à prendre en compte dans l'algorithme d'énumération. Le choix des algorithmes à combiner est aussi une question importante. Nous signalons que combiner efficacement les techniques de recherche locale et systématique correspond au challenge 7 proposé par SELMAN et al. [Selman *et al.* 1997].

La résolution a été pendant longtemps écartée pour des raisons de complexité spatiale (et donc temporelle). Les travaux récents de Laurent Simon et Philippe Chatalic [Chatalic & Simon 2000] et de Dechter [Dechter & Rish 1994] sur la procédure originale de Davis et Putnam (1960), montrent clairement qu'on a tout intérêt à revisiter ce type d'approches. On peut également noter que de nombreuses améliorations ont été obtenues grâce à l'utilisation de différentes formes limitées de résolution dans la procédure DPL (cf. section 2). Dans ce cadre, il me semble intéressant d'essayer de combiner plus fortement la résolution et l'énumération (DPL). Par exemple, on peut tenter de bénéficier des résultats obtenus en logique du premier ordre sur les stratégies de résolution (e.g résolution linéaire, SL-Résolution, etc.).

Pour réaliser une telle intégration, plusieurs questions se posent :

1. Quelle forme de résolution utiliser ? (e.g. résolution bornée, saturation sur les clauses binaires, résolution linéaire, etc.).
2. Quelles informations peut-on déduire ? (e.g. taille des résolvantes, leurs caractéristiques, etc.) et surtout quelles sont les plus intéressantes ? (e.g. binaires, tautologiques [1], etc.).
3. Quand et comment l'utiliser ? (e.g. de manière incrémentale, sur les clauses impliquant la variable de branchement, en limitant le nombre de résolvantes).

En outre, en collaboration avec Laure BRISOUX et Éric GRÉGOIRE, une étude préliminaire a été réalisée autour de l'hybridation de différentes techniques de recherche locale [Brisoux-Devendeville 1999]. Des résultats encourageant ont été obtenus grâce à une intégration de deux techniques de recherche locale aux comportements complémentaires (WSAT [Selman & Kautz 1993] et « *Break-Out method* » de MORISS [Morris 1993]).

Enfin, les résultats récents obtenus par GOMES et al. [Gomes & Selman 1997, Gomes *et al.* 1998] confirment l'importance de ce thème. Ces travaux ont été motivés par une observation simple liée au comportement des différentes techniques de résolution. En effet, sur de nombreux problèmes réels (e.g. planification, ordonnancement), les différentes méthodes de recherche présentent des variations importantes en terme de performances. Autrement dit, des différences significatives en temps d'exécution peuvent être constatées selon que l'on utilise différentes heuristiques sur le même problème, la même heuristique sur différents problèmes, ou encore différentes exécutions d'une même technique de recherche locale. Pour minimiser ce phénomène et améliorer la résolution de ce type de problèmes, ils proposent deux approches, la première est basée sur une intégration efficace de différents algorithmes (appelée « *Algorithm Portfolio Design* ») [Gomes & Selman 1997], la seconde est basée sur une exécution répétée d'un algorithme de recherche systématique, chaque exécution étant bornée par un nombre fixé de retour-arrière (appelé "cutoff") [Gomes *et al.* 1998]. Pour cette dernière approche, une perspective intéressante consiste à étudier les possibilités d'intégration de l'apprentissage des échecs (cf. section 2.4.2.2) pour améliorer les différentes étapes de résolution (les exécutions répétées).

Implémenter efficacement : La résolution de problèmes par ordinateur nécessite non seulement la mise au point d'un algorithme mais également le choix de structures de données permettant de coder efficacement

1. Une résolvante tautologique entre deux clauses binaires traduit une équivalence ou un ou-exclusif entre les deux variables.

les données. Une structure de données est efficace si elle permet de réduire (au moins en pratique) le coût des traitements effectués sur une telle structure. Sans minimiser la portée des heuristiques et des traitement additionnels, le choix d'une bonne structure de données constitue un point important dans la réalisation d'une implémentation efficace.

En outre, il suffit de parcourir les sources des meilleurs implémentations actuelles pour s'apercevoir des nombreuses astuces (certaines sont très discutables) utilisées [2] :

- réduire au maximum le coût des mises à jour en utilisant des structures de données supplémentaires (la complexité en espace étant un paramètre souvent négligé),
- retarder les modifications jusqu'au dernier moment,
- utiliser la programmation par effet de bord et les registres pour les variables locales les plus utilisées,
- privilégier la non-lisibilité du programme en utilisant des instructions les plus proches de la machine (e.g. opérations sur les bits),
- minimiser les allocations mémoires (utilisation de tableaux statiques),
- etc.

Toutes les implémentations que nous avons réalisées jusqu'à présent ne sont aucunement optimisées et n'utilisent pas ces différentes astuces de programmation. Dans ce contexte, la comparaison des méthodes devient malheureusement problématique sachant que le chronomètre est considéré comme l'arbitre suprême. De plus l'implémentation des différents algorithmes dans une même plate-forme est très difficile voire impossible à réaliser. Ce qui nous a conduits avec Bertrand MAZURE à développer une première version de DPL utilisant des structures de données efficaces et différentes astuces que nous avons jugées intéressantes. Nous avons développé une heuristique nouvelle générique et une étape de pré-traitement par résolution bornée très efficace. La prochaine étape consiste à intégrer les différents travaux précédemment réalisés ou en cours de réalisations.

4.1.1.2 Fonctions booléennes générales et propriétés structurelles

Pour se rendre compte des limites de la forme CNF, il suffit de regarder la structure des formules codant les problèmes réels. Je citerai par exemple les instances de circuits proposées par Joao SILVA [Silva & e Silva 1999, Silva & Sakallah 2000], les instances de model checking de A. BIERE [Biere *et al.* 1999], les problèmes de cryptographie de Massacci [Massacci 1999] (ces instances peuvent être téléchargées sur le site de SATLIB [3]). Le comportement des algorithmes développés sur SAT est assez décevant (en particulier la première classe d'algorithmes : les approches prospectives). Ce comportement s'explique en partie par la non prise en compte de la structure particulière de ces problèmes. En effet, de nombreuses clauses ne sont que l'expression d'une formule booléenne générale de la forme $y = f(x_1, x_2, \ldots, x_n)$. Cette constatation simple conduit à reconsidérer le formalisme étendu (cf. section 2.3.2) pour résoudre ce type de problèmes, en ne considérant que des contraintes de la forme $y = f(x_1, \ldots, x_n)$, où f est un connecteur logique ($\wedge$, $\vee$, $\Leftrightarrow$) et éventuellement des formules de cardinalité simple. Le but est d'essayer de mettre en œuvre des méthodes capables d'exploiter une telle structure. Il ne s'agit pas ici d'étendre les algorithmes classiques, mais de trouver des algorithmes plus adaptés et des techniques de simplification spécifiques.

---

2. Dans la description de Satz, ANBULAGAN affirme que l'utilisation des types « char » et « unsigned char » permet d'accélérer Satz de 25% [Anbulagan 1998] !

3. http://www.informatik.tu-darmstadt.de/AI/SATLIB

4.1.1.3 Codage

Améliorer la phase de codage de problèmes réels est une étape indispensable à la résolution efficace du problème proprement dit (voir par exemple, le challenge 8 proposé par Selman [Selman *et al.* 1997] et le thème 1 du GT 1.2 proposé par Daniel LEBERRE et Michel CAYROL [Leberre & Cayrol 1999]). Souvent, la manière de modéliser un problème (e.g. déterminer quels objets représenter et quels types de relations entres les objets) influe considérablement sur la difficulté de résolution du problème. Dans le cadre des applications il est important de considérer à la fois les améliorations au niveau algorithmique et du codage. Un bon codage est un codage qui favorise une résolution efficace. On sait par exemple que la notion de clauses redondantes a une influence sur l'efficacité des algorithmes de résolution (recherche locale et DPL). De nombreux nombres de Ramsey restent actuellement inconnus, on connaît pour certains des majorants et/ou minorants. Résoudre ce type d'instances nécessite en premier lieu de trouver un codage efficace. La notion de symétries fortes introduite dans ma thèse peut contribuer à atteindre un tel objectif.

4.1.2 Autres approches

4.1.2.1 Résolution étendue

Le principe d'extension a été introduit par TSEITIN [Tseitin 1968] : soient $\mathcal{F}$ une formule booléenne, α, β et γ des variables booléennes, où β et γ sont de variables de $\mathcal{F}$ et α une variable (supplémentaire) n'apparaissant pas dans $\mathcal{F}$, alors $\mathcal{F}$ est satisfiable si et seulement si $\mathcal{F} \wedge \{\alpha \Leftrightarrow (\beta \vee \gamma\})$ est satisfiable [4]. Par l'ajout d'une variable supplémentaire pour représenter une formule intermédiaire, on obtient une formule équivalente pour SAT. Cette opération d'ajout est appelée règle d'extension. La résolution étendue est obtenue en augmentant la résolution par la règle d'extension. Des preuves courtes ont été obtenues pour de nombreux problèmes difficiles pour la résolution : problème des pigeons [Cook 1976], formules de Tseitin [Tseitin 1968], problèmes de Ramsey (à deux couleurs) [Krishnamurthy 1985], etc. Ces preuves ont été obtenues grâce à une introduction appropriée de variables supplémentaires représentant des sous-formules. En utilisant ces variables intermédiaires, on évite de construire des résolvantes longues. Ce principe revient à introduire des lemmes pour raccourcir les preuves. Pour exploiter et implémenter une telle méthode, il est nécessaire de répondre à deux questions : « comment introduire de telles variables ? » et « quelle fonction intermédiaire représenter ? ». Ce qui n'est pas du tout évident. En effet, la puissance d'une telle méthode n'est plus à démontrer, mais sa mise en œuvre pratique reste une question difficile. Il suffit de voir le nombre d'auteurs (liste non exhaustive) ayant proposé cette perspective de recherche [Saïs 1993, Castell 1997, Selman 1995, Edwards 1996, Mazure 1999]. Une piste intéressante est d'exploiter les liens entre l'extension et des notions plus exploitables : la notion de clause bloquée [5][Kullmann 1999b] et les NPS [6] [Dubois & Boufkhad 1997].

4.1.2.2 De la structure dans les instances aléatoires ?

Les instances aléatoires $k - SAT$ ne présentent apparemment aucune forme de structure. Une conjecture forte : les instances aléatoires insatisfiables contiennent un noyau minimalement inconsistant contenant des régularités (e.g. symétries). On sait par exemple, que pour un nombre de clauses fixé, plus le nombre de variables augmente plus le seuil tend vers une fonction 0-1. En d'autres termes, le nombre

4. Plus généralement on peut ajouter à $\mathcal{F}$ l'ensemble de clauses codant une fonction booléenne plus générale $\alpha = f(x_1, x_2, \ldots, x_k)$ avec $\alpha \notin var(\mathcal{F})$ et $\forall x \in \{x_1, x_2, \ldots, x_k\}, x \in var(\mathcal{F})$. L'ensemble de clauses ajoutées est appelé extension.

5. Une clause C de $\mathcal{F}$ est dite bloquée si $\exists l \in C, \forall C' \in \mathcal{F}$, tel que $\neg l \in C'$, $Res_l(C, C')$ est tautologique.

6. Une solution s de $\mathcal{F}$ est dite NPS ssi l'interprétation s' obtenue en inversant n'importe quelle variable interprétée à $faux$ contredit $\mathcal{F}$.

de clauses redondantes diminue (pour une étude sur les clauses redondantes [Boufkhad & Roussel 2000, Mazure *et al.* 1998]). Je crois également que lors de la résolution de ce type de problèmes, il n'est pas exclu que les mêmes sous-problèmes à un renommage de variables près soient rencontrés. Ce qui me conduit à proposer ci-dessous une généralisation du théorème de partition du modèle via la symétrie : soient $\mathcal{F}_i$ et $\mathcal{F}_j$ deux ensembles de clauses et σ une application (bijective) de l'ensemble des variables de $\mathcal{F}_i$ dans l'ensemble des variables de $\mathcal{F}_j$, tel que $\sigma(\mathcal{F}_i) \subset \mathcal{F}_j$, alors si $\mathcal{F}_i$ est insatisfiable alors $\mathcal{F}_j$ est insatisfiable. Le problème est que trouver un tel σ est équivalent au problème d'isomorphisme de sous-graphes (un problème NP-Complet).

4.1.2.3 Résolution de SAT par comptage

Une autre technique qu'il est important de revisiter me paraît être la méthode basée sur le comptage des interprétations (non solutions) proposée par IWAMA [Iwama 1987, Iwama 1989]. Cette méthode, appelée aussi algorithme des ensembles indépendants, calcule le nombre d'interprétations supprimées et qui permet de décider de la satisfiabilité de la formule. Il a été montré d'une part que cet algorithme admet un comportement complémentaire aux autres techniques à base d'énumération, et d'autre part que sa complexité en moyenne sur les instances du modèle de probabilité fixé est polynômiale.

4.1.2.4 Formule CNF et représentation en terme de graphes

Il s'agit ici de voir les différentes représentations d'une formule CNF en terme de graphe (e.g. les sommets représentent les clauses et les arêtes représentent par exemple l'existence de résolvantes entre deux clauses, etc.) et d'exploiter ensuite une telle structure. Une voie possible consiste à étudier les possibilités d'adaptation de certains résultats de CSP, traitant de la décomposition de CSP (regroupement en arbres [Dechter & Pearl 1989], ensemble coupe-cycle [Dechter 1990], regroupement cyclique [Jégou 1990]), etc.). Plus généralement la notion de décomposition est liée à la difficulté du problème. En effet, les problèmes difficile tendent à être connexes (e.g. les instances aléatoires, les problèmes structurés), donc difficilement décomposables. Ce qui n'est à mon avis pas le cas des problèmes réels. Cette perspective admet des liens avec l'étude que nous avons commencée sur la généralisation du théorème de partition du modèle (cf. section 2.4.2.1).

4.2 Autour de SAT

4.2.1 Problèmes de satisfaction de contraintes

Dans ce cadre, nous envisageons de poursuivre notre travail dans plusieurs directions détaillées ci-dessous.

4.2.1.1 Filtrages et algorithmes de résolution

Il s'agit dans un premier temps de valider expérimentalement les deux approches proposées (cf. section 3.1). Un algorithme de type MAC est actuellement implémenté. La première approche utilisant une forme partielle de filtrage par consistance de chemin est également intégrée. Il reste à analyser son comportement sur différentes classes de CSP, et plus particulièrement sur les CSP structurés. Pour ce qui est de la seconde approche (« Singleton arc consistance »), une étude est actuellement conduite en collaboration avec Assef CHMEISS et Christian BESSIÈRE.

4.2.1.2 Heuristiques

Nous pensons que des améliorations significatives peuvent encore être obtenues en utilisant des raffinements plus puissants des heuristiques classiques. En effet, sur de nombreux problèmes, les heuristiques traditionnelles ne permettent pas de discriminer suffisamment entre les différentes variables. Ce qui est le cas de l'heuristique de choix de la variable ayant le domaine de taille minimum (« *min dom* »), et à moindre échelle l'heuristique minimisant le rapport taille du domaine sur le degré de la variable (« *min (dom/deg)* »). La prise en compte du degré de la variable dans la fonction de sélection a permis une amélioration significative. Il s'agit d'aller plus loin en prenant en compte le voisinage immédiat de la variable. Les caractéristiques du voisinage peuvent être soit intégrées dans l'heuristique de départ, soit utilisées pour départager les variables (« *tie-breaker* »). L'idée de tenir compte du voisinage vient du fait que l'affectation d'une valeur à une variable a une incidence immédiate sur les variables voisines. Pour montrer l'intérêt de ce travail, nous avons commencé par établir un générateur de CSP réguliers (toutes les variables ont non seulement la même taille du domaine, mais également le même degré). Ce modèle permet de générer des instances beaucoup plus difficiles que celles obtenues par le modèle standard. Ceci s'explique par le fait que les meilleures heuristiques basées sur des paramètres syntaxiques comme la taille des domaines et le degré de la variable ne font aucune différence entre les variables. Au premier niveau de l'arbre, la taille du « *tie-breaker* » est égale au nombre de variables. Cette remarque suggère d'aller vers des fonctions de sélection prenant en compte la sémantique des contraintes.

J'ai essayé récemment de résoudre un CSP aléatoire généré au seuil en utilisant les différentes techniques de SAT sur le codage CNF de l'instance CSP. Le comportement des meilleurs algorithmes (Satz, CSAT, etc.) a été catastrophique, alors que l'algorithme MAC avec une heuristique « *min dom/deg* » a résolu le problème en moins d'une seconde. Ce comportement s'explique par le fait que MAC exploite la structure du graphe de contraintes, ce qui n'est pas le cas des algorithmes de SAT. Ce qui nous a conduits à étudier les pontages entre les heuristiques de SAT et celles des CSP. Des résultats très intéressants ont déjà été obtenus. L'équivalent de « *min dom/deg* » permet de généraliser l'heuristique « *Moms* » utilisée dans le cadre SAT. Il reste à étudier le comportement d'une telle heuristique. On peut déjà affirmer que cette nouvelle heuristique « retrouve » d'une certaine manière la structure du graphe.

4.2.1.3 Extension de la substituabilité de voisinage

La relation de *substituabilité du voisinage* (VS) sur les valeurs d'une variable a été définie dans [Bellicha *et al.* 1994] : pour deux valeurs a et b du domaine de la variable x_i, on dit que a est voisinage substituable par b si et seulement si l'ensemble des valeurs compatibles avec a est inclus dans l'ensemble des valeurs compatibles avec b. La valeur a peut être supprimée du domaine de x_i tout en obtenant un CSP équivalent pour la satisfiabilité. Ce qui permet de définir des techniques de filtrage par VS [Bellicha *et al.* 1994]. L'extension de VS (EVS) que j'ai obtenue est la suivante : a et b sont dits EVS ssi l'ensemble des valeurs compatibles avec a admet une intersection non vide avec l'ensemble des valeurs compatibles avec b sur chacune des variables y_j voisines de la variable x_i. Cette notion ne permet pas de supprimer la valeur a comme pour VS, mais elle peut être utilisée dans le cadre des algorithmes de recherche pour couper des branches de l'arbre. En effet, on peut considérer un seul tuple parmi ceux du produit cartésien des sous-domaines (valeurs appartenant à l'intersection) des variables voisines. Une utilisation dynamique de cette notion, permettra de couper des branches de l'arbre de recherche et d'éviter par conséquent une exploration redondante de l'espace de recherche. Il reste d'une part à valider expérimentalement cette approche et d'autre part de voir si cette extension est équivalente ou non à la notion de NPS (« *Negative Prime Solution* ») proposée par Yacine BOUFKHAD [Boufkhad 1996] dans le cadre SAT. Ce dernier point mérite des investigations supplémentaires. Indépendamment, une étude sur l'extension des NPS au cadre CSP a été entamée par Yacine BOUFKHAD et Assef CHMEISS .

4.2.1.4 Compilation et représentation d'ensembles de solutions

Trouver toutes les solutions d'un CSP est une des questions que l'on peut se poser dans le cadre CSP, et qui admet une utilité dans de nombreuses situations : CSP dynamiques, résolution interactive de CSP, problèmes de configurations, problème de conception interactive, etc. (projet CSPFlex du PRC-IA [CSPFlex 1995]). En plus du problème de calcul d'un tel ensemble de solutions (qui peut être de taille exponentiel dans le pire cas), de nombreux auteurs ont proposé différents modes de représentation utilisant par exemple les automates [Vempaty 1992, Amilhastre 1999], les diagrammes de décision binaires [Bouquet & Jégou 1997], etc. Un sous-groupe de travail autour de ce thème a été proposé par J. ALMIHASTRE, P. JANSSEN et M.C. VILAREM dans le cadre du GT 1.2 du GdR I3.

Dans ce cadre, nous pensons utiliser l'algorithme M(AC+) [Chmeiss & Saïs 2000] (cf. section 3.1) pour calculer une représentation compacte de l'ensemble de solutions. En effet, chaque solution partielle trouvée par M(AC+) représente un sous-ensemble de solutions. De plus, le CSP simplifié par une telle solution est un CSP traitable. Une perspective consiste à étudier les extensions possibles des approches de compilation très largement développées dans le cadre propositionnel aux CSP. Par exemple, les approches développées dans [Boufkhad *et al.* 1997] (cf. section 3.2) peuvent être naturellement étendues au cadre CSP.

4.2.1.5 CSP n-aire

Dans le cas des CSP binaires, de nombreux travaux ont été réalisés autour des filtrages (e.g. consistance d'arc, consistance chemin, singleton arc consistance, etc), des classes traitables, etc. Pour obtenir des algorithmes efficaces dans le cas n-aire [Bessière 1999], une perspective intéressante consiste à étudier ces questions dans le cas n-aires. On peut noter d'ailleurs une nette orientation vers l'étude des CSP n-aires. Comme pour la forme CNF, cette tendance a été motivée par la résolution des problèmes réels dont les contraintes portent généralement sur plus de deux variables.

4.2.2 Procédures de preuve pour les logiques non standard

Les problèmes émanants de logiques non standard sont encore plus difficiles: leur complexité se situe généralement au deuxième niveau de la hiérarchie polynômiale. Cependant, comme pour le problème SAT les applications réelles sont loins de présenter le cas le plus défavorable. Dans la mise en œuvre de méthodes de preuves pour les logiques non standard, il est important de partir d'applications réelles, et de tenir compte des caractéristiques propres de telles applications, quitte à utiliser des techniques heuristiques. L'étude que nous avons réalisée dans ce cadre est un premier pas dans cette direction. Comme je l'avais signalé, c'est grâce à Eric GRÉGOIRE et à Pierre MARQUIS que j'ai fait mes premiers pas dans cette problématique. Il s'agit dans un premier temps d'essayer de progresser dans la compréhension des différentes approches développées dans ce domaine, et d'étudier ensuite les possibilités de mise en œuvre de telles procédures de preuve.

4.3 Applications

J'ai très largement utilisé les termes "applications" et "problèmes réels" dans le cadre de cette synthèse. Outre l'intérêt d'utiliser les instances issues d'applications réelles comme base de tests pour l'évaluation pratique des techniques de résolution mise en œuvre, il me paraît important de considérer les applications depuis la phase de modélisation jusqu'à la résolution en passant par l'étape de codage. En plus du lien étroit entre ces différentes phases, cette approche permet souvent d'exhiber de nouvelles directions de recherche. En effet, les applications permettent de montrer non seulement les limites des méthodes de résolution, mais également les difficultés de représentation.

De plus, de nombreuses applications (e.g. planification, ordonnancement) peuvent être résolues en utilisant différentes techniques issues de différents domaines (e.g. recherche opérationnelle, intelligence artificielle). Il est important de continuer dans ce cadre à étudier les liens entre ces différentes approches et les possibilités d'intégration des techniques issues de l'IA et de la RO. Les résultats obtenus ces dernières années sur ce sujet montrent l'importance de continuer ce type de rapprochement.

Les arguments développés ci-dessus montrent l'importance d'une telle perspective ; je reste cependant conscient des difficultés et du travail nécessaire pour faire un pas significatif dans cette direction. Néanmoins, j'envisage une telle démarche pour mes travaux futurs.

Bibliographie

[Achlioptas *et al.* 2000] Dimitris ACHLIOPTAS, Carla GOMES, Henry KAUTZ, et Bart SELMAN. « Generating Satisfiable Instances ». Dans *AAAI'00*, Austin, Texas, 2000. To appear.

[Aguirre 1992] A. San Miguel AGUIRRE. « Symmetries and the Cardinality Operator ». Dans Bernd NEUMANN, éditeur, *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 18–22, Vienna, Austria, 1992. John Wiley, Sons.

[AI 1996] *Special Volume on Frontier in Problem Solving: Phase Transitions and Complexity*, volume 81, numéro 1 de *Artificial Intelligence*, 1996.

[Akers 1978] B. AKERS. « Binary Decision Diagrams ». *IEEE Transactions on Computers*, 27(6):509–516, 1978.

[Amilhastre 1999] J. AMILHASTRE. « *Représentation par automate d'ensemble de solutions de problèmes de satisfaction de contraintes* ». PhD thesis, Université de Montpelier II, Janvier 1999.

[Anbulagan 1998] ANBULAGAN. « *Hypothèse de contrainte : une explication de la réussite de l' heuristique UP dans la résolution des problèmes de satisfiabilité des expressions booléennes* ». Thèse de doctorat, Université de Technologie de Compiègne, juin 1998.

[André 1993] Pascal ANDRÉ. « *Aspects probabilistes du problème de la satisfaction d'une formule booléenne. Étude des problèmes SAT, #SAT et max-SAT* ». Thèse de doctorat, Université Paris 6, 1993.

[Bahia 1992] Groupe BAHIA. « Etude Comparative de Trois Formalismes en Calcul Propositionnel ». Dans *4èmes journnées nationales du PRC-GDR Intelligence Artificielle*, pages 239–318, Marseille, 19-21 Octobre 1992. Teknea.

[Bahia 1995] Groupe BAHIA. « Etude comparatives de trois formalismes en calcul propositionnel ». Dans TEKNEA, éditeur, *5èmes Journées Nationales du PRC-GDR IA*, pages 125–157, Nancy, Février 1995.

[Barth 1995] P. BARTH. « A Davis-Putnam Based Enumeration Algorithm for Linear pseudo-Boolean Optimization ». Rapport Technique MPI-I-95-2-003, MPI Technical Report, 1995.

[Bayardo & Schrag 1996] R. J. BAYARDO et R. SCHRAG. « Using CSP Look-Back Techniques to Solve Exceptionally Hard SAT Instances ». Dans *Proceedings of the Second International Conference on Principles and Practice of Constraint Programming (CP'96)*, volume 1118, pages 46–60. LNCS, Springer, 1996.

[Bayardo Jr. & Schrag 1997] Roberto J. BAYARDO JR. et Robert C. SCHRAG. « Using CSP Look-Back Techniques to Solve Real-World SAT Instances ». Dans *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI'97)*, pages 203–208, Providence (Rhode Island, USA), juillet 1997.

[Belleannée & Vorc'h 1994] C. BELLEANNÉE et R. VORC'H. « A linear tableau proof for the pigeon-hole formulae using symmetry ». Dans *Proceedings of the Workshop on Theorem*

Proving with Analytic Tableaux and Related Methods, Imperial College - London - UK Oxford, 1994. TR-94/5.

[Bellicha *et al.* 1994] Amit BELLICHA, Michel HABIB, Marie Catherine VILAREM, Christian CAPELLE, et Tibor KÖKÉNY. « CSP Techniques using Partial Orders on Domain Values ». Dans Thomas SCHIEX et Christian BESSIÉRE, éditeurs, *Workshop on Constraint Satisfaction Issues raised by Practical Applications (ECAI'94)*, Amsterdam, août 1994.

[Benhamou & Saïs 1991] B. BENHAMOU et L. SAÏS. « Study of symmetries in propositional calculus ». Dans *International Workshop on Computer Science*, pages 78–102, Annaba, 16-18, Décembre 1991.

[Benhamou & Saïs 1992] Belaïd BENHAMOU et Lakhdar SAÏS. « Theoretical Study of Symmetries in Propositional Calculus and Applications ». Dans *Proceedings of Eleventh International Conference on Automated Deduction (CADE-11)*, volume 607 de *Lecture Notes in Artificial Intelligence*, pages 281–294. Springer Verlag, 1992.

[Benhamou & Saïs 1994] Belaïd BENHAMOU et Lakhdar SAÏS. « Tractability Through Symmetries in Propositional Calculus ». *Journal of Automated Reasoning*, 12:89–102, 1994.

[Benhamou 1994] Belaïd BENHAMOU. « Study of Symmetry in constraint satisfaction problems ». Dans *Proceedings of PPCP'94*, Orcas Island Resario, Washington, 1994.

[Benhamou *et al.* 1992a] B. BENHAMOU, L. SAÏS, et P. SIEGEL. « Dealing with cardinality formuls in propositional calculus ». Dans *International Workshop on Tractable Reasoning (AAA'92)*, pages 6–12, San Jose, California, 12-17 July 1992.

[Benhamou *et al.* 1992b] B. BENHAMOU, L. SAÏS, et P. SIEGEL. « Dealing with symmetries in propositional calculus ». Dans *International Workshop on Tractable Reasoning (AAA'92)*, San Jose, California, 12-17 July 1992.

[Benhamou *et al.* 1994] Belaïd BENHAMOU, Lakhdar SAÏS, et Pierre SIEGEL. « Two Proof Procedures for a Cardinality Based Language in Propositional Calculus ». Dans P. ENJALBERT, E.W. MAYR, et K.W. WAGNER, éditeurs, *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 775 de *Lecture Notes in Computer Science*, pages 71–82, Caen (France), février 1994. Springer Verlag.

[Bennaceur & Plateau 1992] H. BENNACEUR et G. PLATEAU. « FAST: une méthode de résolution du problème linéaire de satisfaction de contraintes ». *Techniques et Sciences Informatique*, 11(3):33–57, 1992.

[Bennaceur & Plateau 1995] H. BENNACEUR et G. PLATEAU. « An exact algorithm for constraint satisfaction problem: Application to logical inference ». *Information Processing Letter*, 48:151–158, 1995.

[Benson & Freuder 1992] B.W. BENSON et E.C. FREUDER. « Interchangeability Preprocessing Can Improve Forward Checking Search ». Dans Bernd NEUMANN, éditeur, *Proceedings of the 10th European Conference on Artificial Intelligence (ECAI'92)*, pages 28–30, Vienna, Austria, 1992. John Wiley, Sons.

[Bessant *et al.* 1998] B. BESSANT, E. GRÉGOIRE, P. MARQUIS, et L. SAÏS. « Combining nonmonotonic reasoning and belief revision: a practical approach ». Dans F. GIUNCHIGLIA, éditeur, *8th Int. Conf. on Artificial Intelligence - Methodology, Systems Applications (AIMSA'98)*, pages 115–128, Sozopol, Bulgaria, september 1998. LNCS 1480.

[Bessant *et al.* 2000] B. BESSANT, E. GRÉGOIRE, P. MARQUIS, et L. SAÏS. « *Frontiers in Belief Revision* », Chapitre Iterated Syntax-Based Revision in a Nonmonotonic Setting. Kluwer Academic Publishers, M.-A. Williams and H. Rott édition, 2000. sous presse.

[Bessière & Régin 1996] C. BESSIÈRE et J-C. RÉGIN. « MAC and combined heuristics: Two reasons to forsake FC (and CBJ?) on hard problems ». Dans *International conference on Constraint programming (CP'96), LNCS 1118*, pages 61–75, 1996.

[Bessière 1999] C. BESSIÈRE. « Non-binary constraints ». Dans *International conference on Constraints Programming (CP'99)*, pages 24–27, 1999. Invited lecture.

[Biere *et al.* 1999] A. BIERE, A. CIMATTI, E. M. CLARKE, M. FUJITA, et Y. ZHU. « Symbolic Model Checking using SAT procedures instead of BDDs ». Dans *Design Automaton Conference, DAC'99*, 1999.

[Billionnet & Sutter 1992] Alain BILLIONNET et Alain SUTTER. « An efficient algorithm fot the 3-satisfiability problem ». *Operations Research Letters*, 12:29–36, 1992.

[Blake 1937] BLAKE. « *Canonical Expressions in Boolean Algebra* ». PhD thesis, University of Chicago, Illinois, USA, 1937.

[Boole 1854] G. BOOLE. *Les lois de la pensée*. Mathesis, 1854.

[Boufkhad & Roussel 2000] Yacine BOUFKHAD et Olivier ROUSSEL. « Redundancy in Random SAT Formulas ». Dans *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI'2000)*, Austin, Texsas, August 2000.

[Boufkhad 1996] Yacine BOUFKHAD. « *Aspects probabilistes et algorithmiques du problème de satisfiabilité* ». Thèse de doctorat, Université de Paris 6, Laboratoire d' Informatique de Paris 6 (LIP6), décembre 1996.

[Boufkhad *et al.* 1997] Yacine BOUFKHAD, Éric GRÉGOIRE, Pierre MARQUIS, Bertrand MAZURE, et Lakhdar SAÏS. « Tractable Cover Compilations ». Dans *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, volume 1, pages 122–127, Nagoya (Japan), août 1997.

[Bouquet & Jégou 1997] F. BOUQUET et P. JÉGOU. « Using obdds to handle dynamic constraints ». *Information Processing Letters*, 62:111–120, 1997.

[Brisoux *et al.* 1998] Laure BRISOUX, Lakhdar SAÏS, et Éric GRÉGOIRE. « Mieux Exploiter les échecs au sein des arbres de recherche à la Davis et Putnam ». Dans *Actes des Quatrièmes Journées Nationales sur la Résolution Pratique des Problèmes NP-Complets (JNPC'98)*, pages 31–39, Nantes (France), mai 1998.

[Brisoux *et al.* 1999] L. BRISOUX, É. GRÉGOIRE, et L. SAÏS. « Improving backtrack search for SAT by means of redundancy ». Dans Z.W. RAS et A. SKOWRON, éditeurs, *11th Int. Symposium on Methodologies for Intelligent Systems (ISMIS'99)*, pages 301–309, Varsovie, Pologne, June 1999.

[Brisoux *et al.* 2000] L. BRISOUX, L. SAIS, et E. GRÉGOIRE. « Recherche locale : vers une exploitation des propriétés structurelles ». Dans *Actes des Sixièmes Journées Nationales sur la Résolution Pratique des Problèmes NP-Complets(JNPC'00)*, pages 243–244, Marseille, 2000.

[Brisoux-Devendeville 1999] L. BRISOUX-DEVENDEVILLE. « *Satisfaisabilité propositionnelle en informatique: Aspects algorithmiques et extensions du formalisme* ». PhD thesis, Université d'Artois, Décembre 1999.

[Bruni & Sassano 2000] Renato BRUNI et Antonio SASSANO. « Detecting Minimaly Unsatisfiable Subformulaein unsatisfiable SAT instances by means of Adaptative Core Search ». Dans *Third Workshop on the Satisfiability Problem (SAT'2000)*, May 14-18 2000.

[Bruynooghe 1980] M. BRUYNOOGHE. « Analysis of Dependencies to Improve the Behaviour of Logic Programs ». Dans *5th Conference on Automated Deduction, CADE'5*, pages 293–305, 1980.

[Bryant 1992] R. BRYANT. « Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams ». *ACM Computing Surveys*, 1992.

[Buno & Buning 1993] M. BUNO et H.K. BUNING. « Report on SAT competition ». *Bulletin of the European Association for Theoritical Computer Science*, 49:143–151, Feb. 1993.

[Carrol 1966] Lewis CARROL. *La logique sans peine*. Herman Paris, 1966.

[Castell & Cayrol 1996] Thierry CASTELL et Michel CAYROL. « Computation of prime implicates and prime implicants by the Davis and Putnam Procedure ». Dans *Proceedings of ECAI'96 Workshop on Advances in Propositional Deduction*, pages 61–64, Budapest (Hungary), août 1996.

[Castell & Cayrol 1997] Thierry CASTELL et Michel CAYROL. « Hidden Gold in Random Generation of SAT Satisfiable Instances ». Dans *International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 372–377, 1997.

[Castell & Fargier 1998] Thierry CASTELL et Hélène FARGIER. « Between SAT and CSP: Propositional Satisfaction Problems and Clausal CSPs ». Dans Henri PRADE, éditeur, *13th European Conference on Artificial Intelligence (ECAI'98), Brighton*, pages 214–218. John Wiley & Sons, 1998.

[Castell 1996] Thierry CASTELL. « Résolution arrière dans la procédure de Davis et Putnam ». Dans *Rencontres Françaises en Intelligence Artificielle (RFIA'96)*, pages 109–117, Rennes (France), 1996.

[Castell 1997] Thierry CASTELL. « *Consistance et déduction en logique propositionnelle* ». Thèse de doctorat, Université Paul Sabatier, Toulouse, janvier 1997.

[C.E. Blair & Lowe 1986] R.G. Jeroslow C.E. BLAIR et J.K. LOWE. « Some results and experiments in programming techniques for propositional logic ». *Comput. Oper. Res.*, 13(5):633–645, 1986.

[Cha & Iwama 1995] Byungki CHA et Kazuo IWAMA. « Performance Test of Local Search Algorithms Using New Types of Random CNF Formulas ». Dans *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 304–310, 1995.

[Chabrier 1997] Jacqueline CHABRIER. « *Programmation Par Contraintes: langages, méthodes et applications sur les domaines entiers et booléens* ». Thèse d'habilitation à diriger des recherches, Université de Bourgogne, décembre 1997.

[Chabrier *et al.* 1991] Jacqueline CHABRIER, Jean-Jacques CHABRIER, et F. TROUSSET. « Résolution Efficace d'un Problème de Satisfaction de Contraintes : le million de reines ». Dans *Onzièmes Journées Internationales sur les Systèmes Experts et leurs Applications*, 1991.

[Chabrier *et al.* 1995] Jacqueline CHABRIER, Vincent JULIARD, et Jean-Jacques CHABRIER. « SCORE(FD/B) an efficient complete local-based serach method for satisfiability problems ». Dans *Proceedings of Constraint Programming Workshop on Solving Really Hard Problems*, pages 25–30, Cassis (France), septembre 1995.

[Chabrier *et al.* 1996] Jacqueline CHABRIER, Jean-Michel RICHER, et Chabrier JEAN-JACQUES. « Resolution of structred SAT problems with SCORE(FD/B) ». Dans *Proceedings of the CP'96 Workshop on Constraint Programming Applications*, Boston, (MA, USA), août 1996.

[Chandru & Hooker 1990] V. CHANDRU et J.N. HOOKER. « Extended Horn sets in Propositional Logic ». *Journal of the Association for Computing Machinery*, 38:205–221, 1990.

[Chandru & Hooker 1999] Vijay CHANDRU et John HOOKER. *Optimization Methods for Logical Inference*. Wiley, 1999.

[Chatalic & Simon 2000] Philippe CHATALIC et Laurent SIMON. « ZRes: The Old Davis-Putnam Procedure meets ZBDD, David ». Dans David MACALLESTER, éditeur, *CADE'17*, pages 449–454, 2000.

[Cheeseman *et al.* 1991] Peter CHEESEMAN, Bob KANEFSKY, et William M. TAYLOR. « Where the Really Hard Problems Are ? ». Dans *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI'91)*, pages 331–337, 1991.

[Chmeiss & Saïs 2000] A. CHMEISS et L. SAÏS. « About Local consistency in Solving CSPs ». Dans *Twelfth IEEE International Conference on Tools with Artificial Intelligence, ICTAI'00*, Vancouver, British Columbia, Canada, November,13-15 2000. IEEE Computer Society. à paraitre.

[Chmeiss 1996] A. CHMEISS. « *Reseaux de contraintes: Algorithmes de Propagation et de Décomposition* ». PhD thesis, Université d'Aix-Marseille I, 1996.

[Chu-Min 2000] Li CHU-MIN. « Integrating Equivalency reasoning into Davis-Putnam procedure ». Dans *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI'2000)*, Austin Texsas, 2000.

[Chvátal & Reed 1992] V. CHVÁTAL et B. REED. « Miks gets some (the odds are on this side) ». Dans *Proceedings of the* 33^{rd} *IEEE Symposium on Foundations of Computer Science*, pages 620–627, 1992.

[Chvátal & Szemerédi 1988] Vašek CHVÁTAL et Endre SZEMERÉDI. « Many hard examples for resolution ». *Journal of the Association for Computing Machinery*, 35(4):759–768, 1988.

[Conforti & Cornuéjols 1992] M. CONFORTI et G. CORNUÉJOLS. « A class of logical inference problems solvable by linear programming ». Dans *Proceedings of FOCS'92*, volume 33, pages 670–675, 1992.

[Cook 1971] S. A. COOK. « The complexity of theorem-proving procedures ». Dans *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, New York (USA), 1971.

[Cook 1976] S.A. COOK. « A short Proof of the Pigeon-Hole Principle Using Extended Resolution ». *SIGACT News*, 8:28–32, 1976.

[Cooper *et al.* 1994] M.C. COOPER, D. A. COHEN, et P. G. JEAVONS. « Characterizing tractable Constraints ». *Artificial Intelligence*, 65:347–361, 1994.

[Crawford & Auton 1993] James M. CRAWFORD et Larry D. AUTON. « Experimental Results on the Crossover Point in Satisfiablity Problems ». Dans *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI'93)*, pages 21–27, 1993.

[Crawford & Auton 1996] James M. CRAWFORD et Larry D. AUTON. « Experimental Results on the Crossover Point in Random 3SAT ». *Artificial Intelligence*, 81(1-2), 1996.

[Crawford & Baker 1994] J.M. CRAWFORD et A.B. BAKER. « Experimental Results on the Application of Satisfiability Algorithms to Scheduling Problems ». Dans *Twelfth National Conference on Artificial Intelligence, AAAI'94*, 1994.

[Crawford 1992] James M. CRAWFORD. « A theorical analysis of reasoning by symmetry in first-order logic ». Dans *Proceedings of the AAAI'92 Workshop on Tractable Reasoning*, San Jose (USA), juillet 1992.

[Crawford 1993] James M. CRAWFORD. « Solving Satisfiability Problems Using a Combination of Systematic and Local Search ». Dans *Working notes of the DIMACS Workshop on Maximum Clique, Graph Coloring, and Satisfiability*, 1993.

[Crawford *et al.* 1996] James CRAWFORD, Matthew L. GINSBERG, Eugene LUCK, et Amitabha ROY. Symmetry-Breaking Predicates for Search Problems. Dans Luigia Carlucci AIELLO, Jon DOYLE, et Stuart SHAPIRO, éditeurs, *Principles of Knowledge Representation and Reasoning (KR'96)*, pages 148–159. Morgan Kaufmann, San Francisco, California, 1996.

[CSPFlex 1995] Projet CSPFLEX. « Autour du problème de satisfaction de contraintes ». Dans *Actes des 5ièmes journées nationales du PRC-GDR Intelligence Artificielle*, Nancy, France, Février 1995. Groupe PRCIA Représentation et traitement de la flexibilité dans les problèmes sous contraintes.

[Cubbada & Mouseigne 1988] C. CUBBADA et M.D. MOUSEIGNE. « *Variantes de l'algorithme de SL-résolution avec retenue d'information* ». Thèse de doctorat, Université d'Aix-Marseille II (GIA Luminy), Marseille (France), 1988.

[Dalal & Etherington 1992] Mukesh DALAL et David W. ETHERINGTON. « A hierarchy of tractable satisfiability problems ». *Information Processing Letters*, 44(4):173–180, 10 décembre 1992.

[Dantsin *et al.* 2000] E. DANTSIN, A. GOERDT, E. A. HIRSCH, et U. SCHÖNING. « Deterministic algorithms for k-SAT based on covering codes and local search ». Dans *ICALP'00*, 2000.

[Davis & Putnam 1960] Martin DAVIS et Hilary PUTNAM. « A Computing Procedure for Quantification Theory ». *Journal of the Association for Computing Machinery*, 7:201–215, 1960.

[Davis *et al.* 1962] Martin DAVIS, George LOGEMANN, et Donald LOVELAND. « A Machine Program for Theorem Proving ». *Journal of the Association for Computing Machinery*, 5:394–397, 1962.

[Debruyne & Bessière 1997] R. DEBRUYNE et C. BESSIÈRE. « Some Practicable Filtering Techniques for the Constraint Satisfaction Problems ». Dans *International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 412–417, 1997.

[Dechter & Pearl 1989] R. DECHTER et J. PEARL. « Tree Clustering for Constraint Networks ». *Artificial Intelligence*, 38:353–366, 1989.

[Dechter & Rish 1994] Rina DECHTER et Irina RISH. « Directional Resolution: the Davis-Putnam procedure revisited ». Dans J. DOYLE, E. SANDEWALL, et P. TORASSI, éditeurs, *Proceedings of the Fourth International Conference on the Principles of Knowledge Representation and Reasoning (KR'94)*, pages 134–145, 1994.

[Dechter 1989] R. DECHTER. « Enhancement Schemes for Constraint Processing: Backjumping, Learning, and cutset Decomposition ». *Artificial Intelligence*, 41:273–312, 1989.

[Dechter 1990] R. DECHTER. « Enhancement Schemes for Constraint-satisfaction problems: Backjumping, Learning and Cutset Decomposition ». *Artificial Intelligence*, 41:273–312, 1990.

[del Val 1994] Alvaro DEL VAL. « On the Relation between the Coherence and Foundations Theories of Belief Revision ». Dans *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI'94)*, pages 909–914, Menlo Park, CA, USA, 1994. AAAI Press.

[Dimacs 1993] DIMACS. « Second Challenge on Satisfiability Testing organized by the Center for Discrete Mathematics and Computer Science of Rutgers University », 1993. `http://dimacs.rutgers.edu/Challenges/`.

[Dubois & Boufkhad 1996] Olivier DUBOIS et Yacine BOUFKHAD. « From very hard doubly balanced SAT formulae to easy unbalanced SAT formulae, variations of the satisfiability threshold ». Dans J.G. DING-ZHU DU et P. PARDALOS, éditeurs, *Proceedings of the DIMACS Workshop on the Satisfiability Problem: Theory and Applications*, mars 1996.

[Dubois & Boufkhad 1997] Olivier DUBOIS et Yacine BOUFKHAD. « A General Upper Bound for the Satisfiability Threshold of Random r-SAT Formulae ». *Journal of Algorithms*, 24(2):395–420, août 1997.

[Dubois & Carlier 1991] Olivier DUBOIS et Jacques CARLIER. « Probabilistic Approach to the Satisfiability Problem ». *Theoretical Computer Science*, 81:65–75, 1991.

[Dubois 1990] Olivier DUBOIS. « On the r,s-SAT satisfiability problem and a conjecture of Tovey ». *Discrete Applied Mathematics*, 26:51–60, 1990.

[Dubois 2000] O. DUBOIS. « Typical random 3-SAT formulae and the satisfiability Threshold ». Dans *Third international workshop on the satisfiability problem (SAT2000)*, 2000.

[Dubois *et al.* 1996] Olivier DUBOIS, Pascal ANDRÉ, Yacine BOUFKHAD, et Jacques CARLIER. « SAT versus UNSAT ». Dans D.S. JOHNSON et M.A. TRICK, éditeurs, *Second DIMACS Challenge, 1993*, DIMACS Series in Discrete Mathematics and Theorical Computer Science, American Mathematical Society, pages 415–436, 1996.

[Dunham & Wang 1976] Bradford DUNHAM et Hao WANG. « Feasible solutions of the tautology problem ». *Annals of Mathematical Logic*, 10:117–154, 1976.

[E. Boros & Kogan 1994] P.L. Hammer E. BOROS et A. KOGAN. « Computational experiments with an exact SAT solver ». Dans *3rd International Symposium On Artificial Intelligence and Mathematics*, Florida, 1994.

[Edwards 1996] Douglas D. EDWARDS. « Research Summary », December 1996. http://www.cs.cornell.edu/home/selman/.

[Ellman 1993] Thomas ELLMAN. « Abstraction via Approximate Symmetry ». Dans *IJCAI'93*, 1993.

[Franco & Paul 1983] J. FRANCO et M. PAUL. « Probabilistic analysis of the Davis-Putnam procedure for solving satisfiability ». *Discrete Applied Mathematics*, 5:77–87, 1983.

[Franco *et al.* 1997] J. FRANCO, J. GU, P.W. PURDOM, et B.W. WAH. *Algorithms for Satisfiability (SAT) Problem: A Survey*. American Mathematical Society, 1997.

[Freeman 1995] Jon William FREEMAN. « *Improvements to Propositional Satisfiability Search Algorithms* ». PhD thesis, University of Pennsylvania, Department of Computer and Information Science, 1995.

[Freuder 1982] E. C. FREUDER. « A Sufficient Condition for Backtrack-free Search ». *Journal of ACM*, 29(1):24–32, 1982.

[Frieze & Suen 1996] Alan FRIEZE et Stephen SUEN. « Analysis of Two Simple Heuristics on a Random Instance of k-SAT ». *Journal of Algorithms*, 20(2):312–355, mars 1996.

[Galil 1977] Z. GALIL. « On the complexity of regular resolution and the Davis-Putnam Procedure ». *Theoretical Computer Science*, 4:23–46, 1977.

[Gallo & Scutellà 1988] Giorgio GALLO et Maria Grazia SCUTELLÀ. « Polynomially solvable satisfiability problems ». *Information Processing Letters*, 29(5):221–227, 24 novembre 1988.

[Gallo & Urbani 1989] Giorgio Gallo GALLO et Giampaolo URBANI. « Algorithms for Testing the Satisfiability of Propositional Formulae. ». *Journal of Logic Programming*, 7(1):45–61, juillet 1989.

[Garey & Johnson 1979] Michael R. GAREY et David S. JOHNSON. *Computers and Intractability: A Guide to the Theory on NP-completness*. A series of books in the Mathematical Sciences. W. H. Freeman and Company, New-York (USA), 1979.

[Génisson & Jégou 1996] R. GÉNISSON et P. JÉGOU. « Davis and Putnam were Already Checking Forward ». Dans *12th European Conference on Artificial Intelligence (ECAI'96)*, pages 180–184, Chichester - New York - Brisbane, août 1996. Wiley.

[Génisson & Rauzy 1996] Richard GÉNISSON et Antoine RAUZY. « Sur les relations algorithmiques des classes polynomiales du problème SAT et des problèmes de satisfaction de contraintes ». Dans *Actes de RFIA'96*, pages 97–107, 1996.

[Génisson & Saïs 1994] Richard GÉNISSON et Lakhdar SAÏS. « Some ideas on random generation of K-SAT instances ». Dans *International Workshop on Experimental Evaluation of Reasoning and Search Methods (AAAI'94)*, pages 91–92, 1994.

[Génisson & Siegel 1994] Richard GÉNISSON et Pierre SIEGEL. « A polynomial method for sub-clauses production ». Dans *Proceedings of Sixth International Conference on Artificila Intelligence: Methodology, Systems, Applications (AIMSA'94)*, pages 25–34, North Holland, 1994.

[Gent & Walsh 1993] Ian P. GENT et Tobby WALSH. « Towards an Understanding of Hill-Climbing Procedures of SAT ». Dans *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI'93)*, pages 28–33, 1993.

[Gent & Walsh 1994] Ian P. GENT et Tobby WALSH. « The SAT phase transition ». Dans *Proceedings of the Eleventh European Conference on Artificial Intelligence(ECAI'94)*, pages 105–109, 1994.

[Ginsberg 1993] Matthew L. GINSBERG. « Dynamic Backtracking ». *Journal of Artificial Intelligence Research (JAIR)*, 1:25–46, 1993.

[Glover 1989] F. GLOVER. « Tabu search - Part I ». *ORSA Journal of Computing*, 1:190–206, 1989.

[Goerdt 1992] A. GOERDT. « A Treshold for Unsatisfiability ». *Lecture Notes in Computer Science*, 629:264–??, 1992.

[Goldberg 1979a] A. GOLDBERG. « Average case complexity of the satisfiability problem ». Dans *4th workshop on Automated Deduction*, pages 1–6, 1979.

[Goldberg 1979b] A. GOLDBERG. « On the complexity of satisfiability problems ». Rapport Technique 16, Courant Computer Science, New York University, 1979.

[Goldberg *et al.* 1983] A. GOLDBERG, P.W. PURDOM, et C.A. BROWN. « Average time analysis of simplified Davis and Putnam Procedures ». *Information Processing Letter*, 15:72–75, 1983.

[Gomes & Selman 1997] Carla P. GOMES et Bart SELMAN. « Algorithm Portfolio Design: Theory vs. Practice ». Dans Dan GEIGER et Prakash Pundalik SHENOY, éditeurs, *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI'97)*, pages 190–197, San Francisco, 1997. Morgan Kaufmann Publishers.

[Gomes *et al.* 1998] Carla P. GOMES, Bart SELMAN, et Henry KAUTZ. « Boosting Combinatorial Search through Randomization ». Dans *Proceedings of the 15th National Conference on Artificial Intelligence, (AAAI'98)*, pages 431–437, 1998.

[Graham *et al.* 1980] R. L. GRAHAM, B. L. ROTHSCHILD, et J. H. SPENCER. *Ramsey Theory*. John Wiley & Sons, 1980.

[Gre] « Greentech Computing Ltd. (UK) ». http://www.greentech-computing.co.uk/.

[Grégoire *et al.* 1998] Éric GRÉGOIRE, Bertrand MAZURE, et Lakhdar SAÏS. « Logically-complete local search for propositional nonmonotonic knowledge bases ». Dans Ilkka NIELELÄ et Torsten SCHAUB, éditeurs, *Proceedings of the KR'98 Workshop on Computational Aspects of Nonmonotonic Reasoning*, numéro 52 dans Series A: Research Reports, pages 37–45. Helsinky University of Technology, Digital Systems Laboratory, 1998. ISSN 0783 5396, ISBN 951 22 4072 6.

[Groote & Warners 2000] Jan Friso GROOTE et Joost P. WARNERS. « The Propositional Formula Checker HeerHugo ». *Journal of Automated Reasoning*, 24, February 2000.

[Gu 1997] J. GU. *Constraint-Based Search*. Combridge University Press, 1997.

[Haken 1985] Armin HAKEN. « The intractability of resolution ». *Theritical Computer Science*, 39:297–308, 1985.

[Hansen & Jaumard 1990] Pierre HANSEN et Brigitte JAUMARD. « Algorithms for the Maximum Satisfiability Problem ». *Journal of Computing*, 22:279–303, 1990.

[Hao & Tétard 1996] Jin-Kao HAO et Laurent TÉTARD. « CH-SAT: A complete Heuristic procedure for satisfiability problem ». Dans *Proceedings of ECAI'96 Workshop on Advances in Propositional Deduction*, pages 27–38, Budapest (Hungary), août 1996.

[Hao 1995] J.K. HAO. « A clausal genetic representation and its related evolutionary procedures for satisfiability problems ». Dans *Int. Conf. on Artificial Neural Nets and Genetic Algorithms*, pages 289–292. Springer-Verlag, 1995.

[Hentenryck & Deville 1990] P.V. HENTENRYCK et Y. DEVILLE. « The cardinality operator: A new logical connective for constraint programming ». Rapport Technique 24, Brown University, Computer Science Departement, 1990.

[Hirsch 2000] E. A. HIRSCH. « SAT Local Search Algorithms: Worst-Case Study ». *Journal of Automated Reasoning (special issue on SAT)*, 24(1/2):127–143, February 2000.

[Hooker & Yan 1999] J. N. HOOKER et H. YAN. « Tight representation of logical constraints as cardinality rules ». *Mathematical Programming*, 85:363–377, 1999.

[Hooker 1988] J.N. HOOKER. « A quantitative approach to logical inference ». *Decision Support Systems*, 7(1):1–7, 1988.

[Hooker 2000] John HOOKER. *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. Wiley, 2000. sous press.

[Hoos & Stützle 2000] Holger H HOOS et Thomas STÜTZLE. « SATLIB: An Online Ressource for Research on SAT ». SAT 2000, Ian Gent, Hans van Maaren, and Toby Walsh, editors, IOS Pres, 2000. http://aida.intellektik.informatik.tu-darmstadt.de/SATLIB/.

[Iwama 1987] Kazuo IWAMA. « Complementary Approaches to CNF Boolean Equations ». *PERSPEC: Perspectives in Computing*, 15, 1987.

[Iwama 1989] Kazuo IWAMA. « CNF-satisfiability test by counting and polynomial average time ». *SIAM Journal on Computing*, 18(2):385–391, avril 1989.

[Jeannicot *et al.* 1988] S. JEANNICOT, Laurent OXUSOFF, et Antoine RAUZY. « Évaluation sémantique en calcul propositionnel : une propriété de coupure pour rendre plus efficace la procédure de Davis et Putnam ». *Revue d'Intelligence Artificielle*, 2(1):41–60, 1988.

[Jeroslow & Wang 1990] Robert G. JEROSLOW et Jinchang WANG. « Solving Propositional Satisfiability Problems ». *Annals of Mathematics and Artificial Intelligence*, 1:167–187, 1990.

[Jégou 1990] Ph. JÉGOU. « Cyclic-Clustering: a Compromise Between Tree-Clustering and the Cycle-Cutset Method for Improving Search Efficiency ». Dans *ECAI'90*, pages 369–371, Stockholm, 1990.

[Jégou 1993] P. JÉGOU. « Decomposition of domains based on the structure of finite constraints satisfaction problems ». Dans *AAAI'93*, pages 731–736, 1993.

[Johnson 1999] Geroge JOHNSON. « Separating Insolvable and Difficult ». The New York Times on the Web, July 13 1999. http://www.nytimes.com/.

[Kamath *et al.* 1994] Anil KAMATH, Rajeev MOTWANI, Krishna PALEM, et Paul SPIRAKIS. « Tail Bounds for Occupancy and the Satisfiability Threshold Conjecture ». Dans *35th Annual Symposium on Foundations of Computer Science, IEEE*, pages 592–603, Santa Fe, New Mexico, 20–22 novembre 1994.

[Kautz & Selman 1992] H. KAUTZ et B. SELMAN. « Planning as satisfiability ». Dans *European Conference on Artificial Intelligence (ECAI'92)*, pages 359–363, Vienna, Austria, 1992.

[Kim & Zhang 1994] Sun KIM et Hantao ZHANG. « ModGen: Theorem Proving by Model Generation ». Dans *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI'94)*, pages 162–167, Menlo Park, CA, USA, 1994. AAAI Press.

[Kowalski & Kuehner 1971] R.A. KOWALSKI et D. KUEHNER. « Linear Resolution with Selection Function ». *Artificial Intelligence*, 2:227–260, 1971.

[Krishnamurthy 1982] B. KRISHNAMURTHY. « *Examples of hard tautologies and Word-Case complexity results in propositional calculus* ». PhD thesis, Dept. of Computer and Information Science, University of Massachussets, 1982.

[Krishnamurthy 1985] B. KRISHNAMURTHY. « Short Proofs for Tricky Formulas ». *Acta Informatica*, 22:253–275, 1985.

[Kullmann 1999a] Oliver KULLMANN. « New methods for 3-SAT decision and worst-case analysis ». *Theoretical Computer Science*, 223(1-2):1–72, July 1999.

[Kullmann 1999b] Oliver KULLMANN. « On a generalization of extended resolution ». *Discrete Applied Mathematics*, 96-97(1-3):149–176, 1999.

[Lang & Marquis 1998] J. LANG et P. MARQUIS. « Complexity Results for Independence and Definability in Propositional Logic ». Dans *6th International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 356–367, Trento, 1998.

[Larrabee & Tusji 1993] T. LARRABEE et Y. TUSJI. « Evidence for satisfiability threshold for rand 3CNF formalas ». Dans H.E. HIRSH, éditeur, *Proceedings of Spring Symposium on Artificial Intelligence and NP-Hard Problems*, pages 112–118, Stanford (CA, USA), 1993.

[Larrabee 1992] T. LARRABEE. « Efficient generation of test patterns unsing boolean satisfiability ». *IEEE Transaction on CAD*, 11:4–15, 1992.

[Leberre & Cayrol 1999] Daniel LEBERRE et Michel CAYROL. « Influence du codage en calcul propositionnel », 1999. Thème 1 du Groupe de Travail 1.2 "axe algorithme" (GdR I3).

[Li & Anbulagan 1997] Chu Min LI et ANBULAGAN. « Heuristics Based on Unit Propagation for Satisfiability Problems ». Dans *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 366–371, Nagoya (Japan), août 1997.

[Liberatore & Schaerf 1996] Paolo LIBERATORE et Marco SCHAERF. « The Complexity of Model Checking for Belief Revision and Update ». Dans *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 556–561, Menlo Park, 1996. AAAI Press / MIT Press.

[Lobjois & Lemaître 1997] Lionel LOBJOIS et Michel LEMAÎTRE. « Coopération entre méthodes complètes et incomplètes pour la résolution de (V)CSP : une tentative d'inventaire ». Dans *Actes des Troisièmes Journées Nationales sur la Résolution Pratique de Problèmes NPè-complets (JNPC'97)*, pages 67–73, Rennes (France), avril 1997. PRC-GDR IA – Universitè de Rennes 1.

[Loveland 1978] D. LOVELAND. *Automated Theorem Proving: A Logical Basis*. North Holland, 1978.

[Makinson & Gärdenfors 1991] David MAKINSON et Peter GÄRDENFORS. Relations between the Logic of Theory Change and the Nonmonotonic Logic. Dans André FUHRMANN et M. MORREAU, éditeurs, *The Logic of Theory Change*, pages 185–205. Springer-Verlag, Berlin, 1991.

[Marquis 1995] Pierre MARQUIS. « Knowledge compilation using theory prime implicates ». Dans *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 837–843, 1995.

[Massacci & Marraro 2000] Fabio MASSACCI et Laura MARRARO. « Logical cryptanalysis as a SAT-problem: Encoding and analysis of the u.s. Data Encryption Standard ». *Journal of Automated Reasoning*, 2000.

[Massacci 1999] F. MASSACCI. « Using Walk-SAT and Rel-SAT for Cryptographic Key Search ». Dans *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 290–295, 1999.

[Mazure 1995] Bertrand MAZURE. « Expérimentations, analyse et amélioration des méthodes de résolution du problème SAT ». Mémoire de dea, CRIL – Université d'Artois, Centre de Recherche en Informatique de Lens (Faculté Jean Perrin), juillet 1995.

[Mazure 1999] Bertrand MAZURE. « *De la Satisfaisabilité à la Compilation de Bases de Connaissances Propositionnelles* ». Thèse de doctorat, Université d'Artois, Centre de Recherche en Informatique de Lens (Faculté Jean Perrin), janvier 1999.

[Mazure *et al.* 1996] Bertrand MAZURE, Lakhdar SAÏS, et Éric GRÉGOIRE. « Detecting Logical Inconsistencies ». Dans *Proceedings of Mathematics and Artificial Intelligence Symposium*, pages 116–121, Fort Lauderdale (FL USA), janvier 1996.

[Mazure *et al.* 1997a] Bertrand MAZURE, Lakhdar SAÏS, et Éric GRÉGOIRE. « A comparison of Two Approaches to Inconsistency Detecting ». Dans *European Symposium on Intelligent Techniques*, Bari (Italy), mars 1997.

[Mazure *et al.* 1997b] Bertrand MAZURE, Lakhdar SAÏS, et Éric GRÉGOIRE. « An Efficient Technique to ensure the Logical Consistency of Interacting Knowledge Bases ». *International Journal of Cooperative Information Systems*, 6(1):27–36, 1997.

[Mazure *et al.* 1998] Bertrand MAZURE, Lakhdar SAÏS, et Éric GRÉGOIRE. « Boosting Complete Techniques thanks to local search methods ». *Annals of Mathematics and Artificial Intelligence*, 22:319–331, 1998.

[McAllester 1980] D.A. MCALLESTER. « An Outlook on Truth Maintenance ». AI Memo551, August 1980. MIT AI Laboratory.

[McAllester *et al.* 1997] David A. MCALLESTER, Bart SELMAN, et Henry A. KAUTZ. « Evidence for Invariants in Local Search ». Dans *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI'97)*, pages 321–326, août 1997.

[Minton *et al.* 1990] Steven MINTON, Marc D. JOHNSTON, Andrew B. PHILIPS, et Philip LAIRD. « Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method ». Dans *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI'90)*, 1990.

[Mitchell *et al.* 1992] David MITCHELL, Bart SELMAN, et Hector J. LEVESQUE. « Hard and Easy Distributions of SAT Problems ». Dans *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI'92)*, pages 459–465, 1992.

[Moinard 1994] Y. MOINARD. « Revision and nonmonotonicity ». *Int. Journ. of Intelligent Systems*, 9, 1994.

[Monien & Speckenmeyer 1985] B. MONIEN et E. SPECKENMEYER. « Solving satisfiability in less than 2^n steps ». *Discrete Applied Math.*, 10:287–295, 1985.

[Morris 1993] P. MORRIS. « The Break Out Method For Escaping From Local Minima ». Dans *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI'93)*, pages 40–45, 1993.

[Nebel 1992] B. NEBEL. Syntax-Based Approaches to Belief Revision. Dans P. GÄRDENFORS, éditeur, *Belief Revision*, Cambridge Tracts in Theoretical Computer Science 29, pages 52–88. Cambridge University Press, Cambridge, UK, 1992.

[Newell 1982] A. NEWELL. « The knowledge level ». *Artificial Intelligence*, 18:87–127, 1982.

[Oxusoff & Rauzy 1989] Laurent OXUSOFF et Antoine RAUZY. « *L'évaluation sémantique en calcul propositionnel* ». Thèse de doctorat, Groupe d'Intelligence Artificielle Université d'Ais-Marseille II, Faculté de Luminy, Marseille (France), janvier 1989.

[Papini 1991] O. PAPINI. « Revision in Propositional Calculus ». Dans Rudolf KRAUSE et Pierre SIEGEL, éditeurs, *Proceedings of Symbolic and Quantitative Approaches to Uncertainty (ECSQAU'91)*, volume 548 de *LNCS*, pages 272–276, Berlin, Germany, octobre 1991. Springer.

[Polya & Read 1987] G. POLYA et R.C. READ. *Combinatorial Enumeration of Groups, Graphs, and Chemical Compounds*. Springer-Verlag, 1987.

[Pretolani 1993] D. PRETOLANI. « Solving satisfiability problems: An algorithm implementation challenge? ». Dans *Working notes, 2nd DIMACS Challenge*, 1993. Extended abstract.

[Boy de la Tour & Demri 1995] T. BOY DE LA TOUR et S. DEMRI. « On the complexity of Extending Ground Resolution with Symmetry Rules ». Dans *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 289–295, 1995.

[Prover] PROVER. « A commercially packaged proof engine ». Compagny founded in 1989. http://www.prover.com/.

[Puget 1993] J.-F. PUGET. « On the Satisfiability of Symmetrical Constrainted Satisfaction Problems ». Dans J. KOMOROWSKI et Z. W. RAŚ, éditeurs, *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems (ISMIS'93)*, volume 689 de *LNAI*, pages 350–361, Trondheim, Norway, juin 1993.

[Purdom 1990] P. PURDOM. « A survey of average time analyses of satisfiability algorithms ». *Information Processing*, 13(4), 1990.

[Ramsey 1930] F.P. RAMSEY. « On a Problem of Formal Logic ». Dans *London Math. Soc.*, pages 264–286, 1930.

[Rauzy 1994] Antoine RAUZY. « On the Complexity of the Davis & Putnam's Procedure on Some Polynomial Sub-Classes of SAT ». Rapport Interne 806-94, LaBRI, Université de Bordeaux I, 1994.

[Rauzy 1995a] Antoine RAUZY. « On the Random Generation of 3-SAT Instances ». Rapport Interne 1060-95, LaBRI, Université de Bordeaux I, 1995.

[Rauzy 1995b] Antoine RAUZY. « Polynomial restrictions of SAT: What can be done with an efficient implementation of the Davis and Putnam's procedure. ». Dans

U. MONTANARI et F. ROSSI, éditeurs, *Proceedings of the International Conference of Principle of Constraint Programming, CP'95*, volume 976 de *Lecture Notes in Computer Science*, pages 515–532. Springer Verlag, 1995.

[Rauzy *et al.* 1999] A. RAUZY, L. SAÏS, et L. BRSOUX. « Calcul Propositionnel : Vers une extension du formalisme ». Dans *JNPC'99*, pages 189–198, Lyon, 1999.

[Reiter & Mackworth 1989] R. REITER et A. MACKWORTH. « A logical framework for depiction and image interpretation ». *Artificial Intelligence*, 43(2):125–155, 1989.

[Régin & eds 1998] J.C. RÉGIN et W. Nuijtens EDS, éditeurs. *International Workshop on Non-binary constraints (ECAI'98)*, Brighton, 1998.

[Robinson 1965] J.A. ROBINSON. « A machine-oriented logic based on the resolution principle ». *Journal of the Association for Computing Machinery*, 12:23–41, 1965.

[Sabin & Freuder 1994] D. SABIN et E. FREUDER. « Contradicting conventional wisdom in constraint satisfaction ». Dans *ECAI'94*, pages 125–129, 1994.

[Saïs 1993] Lakhdar SAÏS. « *Étude des Symétries et de la Cardinalité en Calcul Propositionnel : Application aux algorithmes sémantiques* ». Thèse de doctorat, Université de Provence, Marseille (France), février 1993.

[Saïs 1994a] L. SAÏS. « Characterization of the set of models by means of symmetries ». Dans *Proceedings of the second Workshop on the Principles and Practice of Contraint Programming (PPCP'94)*, Orcas Island, Washington USA, May 2-4 1994.

[Saïs 1994b] L. SAÏS. « A Computational Study of DP with Symmetry on Hard Satisfiability Problems ». Dans *International Workshop On Experimental Evaluation of Reasoning and Search Methods (AAAI'94)*, pages 52–56, 1994.

[Saïs 1994c] L. SAÏS. « Finding non isomorphic solutions ». Dans P. JORRAND et V. SGUREV, éditeurs, *6th International Conference on AI : Methodology, Systems and Application (AIMSA'94)*, pages 35–44, Sofia Bulgaria, Sept 21-24 1994.

[Schiermeyer 1983] I. SCHIERMEYER. « Solving 3-Satisfiability in less than $O(1.579^n)$ steps ». *Lecture Notes in Computer Science*, 702:379–394, 1983.

[Schiermeyer 1996] I. SCHIERMEYER. « Pure literal lookahead: an $O(1.497^n)$ 3-Satisfiability algorithm ». Dans *Workshop on Satisfiability*, pages 63–72, Universita delgi Studi, Siena, 1996.

[Schiex & Verfaillie 1993] T. SCHIEX et G. VERFAILLIE. « Nogood Recording for Static and Dynamic Constraint Satisfaction Problems ». Dans *International Conference on Tools with Artificial Intelligence (ICTAI'93)*, pages 48–55, 1993.

[Schlipf *et al.* 1995] John S. SCHLIPF, Fred S. ANNEXSTEIN, John V. FRANCO, et R. P. SWAMINATHAN. « On finding solutions for extended Horn formulas ». *Information Processing Letters*, 54(3):133–137, 12 mai 1995.

[Schrag 1996] Robert C. SCHRAG. « Compilation for critically constrained knowledge bases ». Dans *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI'96)*, pages 510–515, juillet 1996.

[Selman & Kautz 1991] Bart SELMAN et Henry A. KAUTZ. « Knowledge compilation using Horn approximations ». Dans *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI'91)*, pages 904–909, 1991.

[Selman & Kautz 1993] Bart SELMAN et Henry A. KAUTZ. « Domain-Independent Extensions to GSAT: Solving Large Structured Satisfiability Problems ». Dans *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'93)*, pages 290–295, 1993.

[Selman 1995] Bart SELMAN. « Stochastic Search and Phase Transitions: AI Meets Physics ». Dans Chris S. MELLISH, éditeur, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 998–1002, San Mateo, 1995.

[Selman *et al.* 1992] Bart SELMAN, Hector J. LEVESQUE, et David MITCHELL. « GSAT: A New Method for Solving Hard Satisfiability Problems ». Dans *Proceedings of the*

Tenth National Conference on Artificial Intelligence (AAAI'92), pages 440–446, 1992.

[Selman *et al.* 1994] Bart SELMAN, Henry A. KAUTZ, et B. COHEN. « Noise strategies for improving local search ». Dans *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI'94)*, pages 337–343, 1994.

[Selman *et al.* 1997] Bart SELMAN, Henry A. KAUTZ, et David A. MCALLESTER. « Computational Challenges in Propositional Reasoning and Search ». Dans *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, volume 1, pages 50–54, Nagoya (Japan), août 1997.

[Shannon 1938] C.E. SHANNON. « A symbolic analysis of relay and switching circuits ». *Transactions AIEE*, 57:713–719, 1938.

[Sheeran & Stalmarck 2000] Mary SHEERAN et Gunnar STALMARCK. « A Tutorial on Stalmarck's Proof Procedure for Propositional Logic ». *Formal Methods in System Design, Kluwer Academic Pu*, 16(1):23–58, January 2000.

[Siegel 1987] Pierre SIEGEL. « *Représentation et Utilisation de la connaissance en calcul propositionnel* ». Thèse d'état, Université de Provence, GIA–Luminy, Marseille (France), 1987.

[Siegel 1990] Pierre SIEGEL. « Communication personnelle », 1990.

[Silva & Sakallah 1996] João P. Marques SILVA et Karem A. SAKALLAH. « GRASP – A New Search Algorithm for Satisfiability ». Dans *IEEE/ACM International Conference on Computer-Aided Design*, November 1996.

[Silva & Sakallah 2000] João P. Marques SILVA et Karem A. SAKALLAH. « Boolean Satisfiability in Electronic Design Automation ». Dans *in Proceedings of the IEEE/ACM Design Automation Conference, DAC'00*, June 2000.

[Silva & e Silva 1999] João P. Marques SILVA et Luís Guerra e SILVA. « Algorithms for Satisfiability in Combinational Circuits Based on Backtrack Search and Recursive Learning ». Dans *Proceedings of the XII Symposium Integrated Circuits and Systems Design (SBCCI'99)*, 1999.

[Slaney 1994] John SLANEY. « The crisis in finite mathematics: Automated reasoning as cause and cure ». Dans Alan BUNDY, éditeur, *Proceedings of the 12th International Conference on Automated Deduction*, volume 814 de *LNAI*, pages 1–13, Berlin, 1994.

[Slaney *et al.* 1993] J. SLANEY, M. FUJITA, et M. STICKEL. « Automated Reasoning and Exhaustive Search: Quasigroup Existence Problems ». *Computers and Mathematics with Applications*, 1993.

[Smith *et al.* 2000] Barbara SMITH, Kostas STERGIOU, et Toby WALSH. « Using Auxiliary Variables and Implied Constraints to Model Non-binary Problems ». Dans *AAAI'2000*, 2000.

[Sosic & Gu 1991] R. SOSIC et J. GU. « 3 000 000 queens in less than one minute ». *Sigart Bulletin (AI), ACM press*, 2(2):22–24, 1991.

[Stallman & Sussman 1977] R.M. STALLMAN et G.J. SUSSMAN. « Forward Reasoning and Dependency-Directed backtraking in a System for Computer AIded Circuit Analysis ». *Artificial Intelligence*, 9:135–196, October 1977.

[Tovey 1984] Craig A. TOVEY. « A Simplified NP-complete Satisfiability Problem ». *Discrete Applied Mathematics*, 8:85–89, 1984.

[Tseitin 1968] G.S. TSEITIN. « On the complexity of derivations in the propositional calculus ». Dans H.A.O. SLESENKO, éditeur, *Structures in Constructives Mathematics and Mathematical Logic, Part II*, pages 115–125, 1968.

[Uribe & Stickel 1994] T. E. URIBE et M. E. STICKEL. « Ordered Binary Decision Diagrams and the Davis-Putnam Procedure ». *Lecture Notes in Computer Science*, 845:34–??, 1994.

[Urquhart 1987] A. URQUHART. « Hard Examples for Resolution ». *ACM*, 34:209–219, 1987.

[Van Gelder & Tsuji 1996] Allen VAN GELDER et Yumi K. TSUJI. « Satisfiability Testing with More Reasoning and Less Guessing ». Dans D.S. JOHNSON et M.A. TRICK, éditeurs, *Second DIMACS Challenge*, DIMACS Series in Discrete Mathematics and Theorical Computer Science, American Mathematical Society, pages 559–866, 1996.

[van Gelder 1999] van GELDER. « Complexity Analysis of Propositional Resolution with Autarky Pruning ». *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 96, 1999.

[Vempaty 1992] Nageshwara Rao VEMPATY. « Solving Constraint Satisfaction Problems Using Finite State Automata ». Dans William SWARTOUT, éditeur, *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI'92)*, pages 453–458, San Jose, CA, juillet 1992. MIT Press.

[Vidal 2000] V. VIDAL. « Contribution à la planification par compilation de plans ». Rapport Technique 00/03-R, IRIT, Université Paul Sabatier, 2000.

[Vlach 1993] F. VLACH. « Simplification in a Satisfiability Checker for VLSI Applications ». *Journal of Automated Reasoning*, 10:115–136, 1993.

[Walser 1997] Joachim P. WALSER. « Solving Linear pseudo-Boolean Constraint Problems with Local Search ». Dans *Proceedings of the 14th National Conference on Artificial Intelligence and 9th Innovative Applications of Artificial Intelligence Conference (AAAI'97)*, pages 269–274, Menlo Park, 1997.

[Walser 1999] Joachim Paul WALSER. *Integer optimization by local search: a domain-independent approach*, volume 1637 de *Lecture Notes in Computer Science and Lecture Notes in Artificial Intelligence*. Springer-Verlag Inc., New York, NY, USA, 1999.

[Zhang & Hsiang 1994] H. ZHANG et J. HSIANG. « Solving open quasigroup problems by propositional reasoning ». Dans *International Computer Symposium*, Hsinchu, Taiwan, 1994.

[Zhang & Zhang 1995] J. ZHANG et H. ZHANG. « SEM: a System for Enumerating Models ». Dans *International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 11–18, Montreal, August 11-18 1995.

[Zhang 1997] H. ZHANG. « SATO: An Efficient Propositional Prover ». Dans *International Conference on Automated Deduction, CADE'97*, 1997.

[Zhang *et al.* 1997] Dongmo ZHANG, Shifu CHEN, Wujia ZHU, et Hongbing LI. « Nonmonotonic Reasoning and Multiple Belief Revision ». Dans *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, volume 1, pages 95–101, 1997.