
Une vision trivaluée pour les problèmes SAT et MAX-SAT

Frédéric Lardeux Frédéric Saubion Jin-Kao Hao

LERIA, Université d'Angers,
2 Bd Lavoisier,
49045 Angers Cedex 01

{lardeux, saubion, hao}@info.univ-angers.fr

Résumé

Dans cet article nous proposons un nouveau cadre de résolution pour les problèmes SAT et MAX-SAT. Ce cadre introduit la troisième valeur de vérité "indéterminé" dans le but d'améliorer l'efficacité de la résolution. Nous avons adapté l'algorithme de recherche Tabou ainsi que Walksat avec ce nouveau cadre de résolution. Des résultats prometteurs ont été obtenus et montrent l'intérêt de notre approche.

Abstract

The aim of this paper is to propose a new resolution framework for the SAT and MAX-SAT problems which introduces a third truth value *undefined* in order to improve the resolution efficiency. Using this framework, we have adapted the classic algorithms Tabu Search and Walksat. Promising results are obtained and show the interest of our approach.

1 Introduction

Le problème de satisfiabilité (SAT) [5] consiste à trouver une affectation satisfaisant une expression Booléenne. Une instance de ce problème SAT est donc définie par un ensemble de variables Booléennes $\mathcal{X} = \{x_1, \dots, x_n\}$ et une formule Booléenne $\phi: \mathcal{B}^n \rightarrow \mathcal{B}$. Cette formule ϕ est supposée être en forme normale conjonctive (CNF) (i.e., une conjonction de clauses où une clause est une disjonction de littéraux¹). La formule est dite satisfiable s'il existe une affectation (i.e., une fonction $A: \mathcal{X} \rightarrow \mathcal{B}^n$) satisfaisant ϕ et non satisfiable dans le cas contraire. L'espace de recherche \mathcal{S} correspond à l'ensemble de toutes les affectations possibles. ϕ est évidemment satisfaite si toutes ses clauses

sont satisfaites. Dans ce contexte, le problème de la satisfiabilité maximale (MAX-SAT) correspond à la minimisation du nombre de clauses fausses.

Deux classes de méthodes sont utilisées pour résoudre les problèmes SAT et MAX-SAT : les méthodes exactes et les méthodes approchées.

Les méthodes exactes sont capables de trouver toutes les solutions pour une instance donnée ou, s'il n'y en a pas, de prouver sa non satisfiabilité. Ces méthodes sont généralement basées sur la procédure Davis-Putnam-Logemann-Loveland [3] qui explore un arbre binaire de recherche en construisant incrémentalement les affectations. Elles fournissent de très bons résultats mais ne sont pas appropriées pour le problème MAX-SAT. D'autres méthodes exactes, généralement basées sur un algorithme de Branch and Bound (B&B) [2, 1, 18], ont été développées pour travailler sur le problème MAX-SAT mais leurs performances sont souvent limitées pour les instances de grandes tailles.

Les méthodes approchées sont principalement basées sur la recherche locale et les algorithmes évolutionnaires. Dans ce papier, nous nous intéressons plus particulièrement aux algorithmes de recherche locale qui ont été largement étudiés par la communauté SAT [15, 14, 17, 10, 8]. L'exploration de l'espace de recherche est réalisée par des mouvements successifs d'un élément à un autre grâce à des heuristiques spécifiques. Le but est de minimiser, sur l'espace de recherche, une fonction retournant le nombre de clauses fausses. Ces algorithmes sont donc naturellement conçus pour le problème MAX-SAT et permettent de manipuler des instances de grande taille.

L'efficacité de tels algorithmes est fortement liée à

1. Un littéral est une variable ou sa négation.

leurs capacités à exploiter en détails des portions de l'espace de recherche quand cela est nécessaire (i.e., intensifier la recherche) et d'explorer de manière plus large l'espace de recherche en se déplaçant entre différents sous-espaces prometteurs (i.e., diversifier la recherche). Ces aspects sont contrôlés grâce à des paramètres qui assurent un bon compromis entre intensification et diversification. Actuellement, ces deux notions ne sont pas si clairement définies. Par conséquent, le réglage de ces paramètres reste un des points cruciaux pour le bon comportement de tels algorithmes.

Nous pouvons remarquer que les méthodes exactes et approchées n'utilisent pas la même représentation pour leur espace de recherche. Leurs hybridations [11, 9, 12, 7, 13] ne sont pas évidentes puisque les méthodes exactes travaillent avec des affectations partielles incrémentalement complétées par un processus de résolution alors que les méthodes approchées explorent l'ensemble de toutes les affectations possibles. Une affectation partielle est une affectation où des variables ne sont pas encore évaluées et peut donc être considérées comme un ensemble d'affectations complètes. D'un point de vue recherche locale, l'utilisation de ces affectations partielles peut mener à de nouveaux processus de diversification et d'intensification. En effet, la diversification de la recherche peut être faite en se déplaçant vers un plus grand ensemble d'affectations complètes, c'est à dire en rendant non définie une variable déjà évaluée alors que l'instanciation de variables non évaluées d'une affectation partielle peut être considérée comme une manière de se recentrer sur un plus petit espace de recherche.

L'objectif de cet article est de définir un modèle uniforme pour ces deux types d'affectations dans le but de définir précisément et d'étudier les mécanismes fondamentaux de la recherche locale pour les problèmes SAT et MAX-SAT. Ce cadre de travail nous permet de proposer un nouveau schéma de recherche locale qui fournit un contrôle uniforme de la recherche et qui peut être intégré dans des algorithmes bien connus de recherche locale: la recherche Tabou et Walksat.

2 Un Cadre de Résolution Trivalué

Nous proposons un nouveau cadre de résolution pour les problèmes SAT et MAX-SAT qui introduit une troisième valeur de vérité *indéterminé* dans le but d'améliorer l'efficacité de la résolution. En utilisant ce cadre de travail, les méthodes de recherche locale seront étendues afin de prendre en compte les affectations partielles et, en conséquence, plusieurs notions et règles logiques devront être redéfinies.

2.1 Recherches Locales Standard

Dans le contexte classique de SAT et MAX-SAT, l'espace de recherche \mathcal{S} est l'ensemble de toutes les affectations complètes. La fonction objective à maximiser correspond au nombre de clauses satisfaites donné par la fonction *eval_standard*:

$$\begin{aligned} eval_standard: \mathcal{S} &\rightarrow \mathbb{N} \\ A &\mapsto |\{c | sat(A, c) \wedge c \in \phi\}| \end{aligned}$$

où $sat(A, c)$ signifie que la clause c est satisfaite par l'affectation $A \in \mathcal{S}$ et $|E|$ est la cardinalité de l'ensemble E .

Les mouvements sont clairement les flips possibles des valeurs d'une affectation donnée. Le flip d'une variable i dans une affectation A est le changement de sa valeur de vérité (V (*vrai*) à F (*faux*) ou F à V). Le meilleur flip est sélectionné grâce à la fonction de choix *choix_standard* qui retourne la variable dont le flip fournit la meilleure amélioration (i.e., maximise le nombre de clauses fausses qui deviennent vraies après le flip moins le nombre de clauses vraies qui deviennent fausses).

Un algorithme de recherche locale naïf pour SAT et MAX-SAT consiste à sélectionner le meilleur mouvement à chaque pas (en utilisant *choix_standard*). Il s'arrête soit quand une solution est trouvée soit quand un nombre maximum de pas de recherche est atteint. Ensuite, il retourne la meilleure affectation trouvée (par rapport à *eval_standard*). Des heuristiques spécifiques de contrôle peuvent être ajoutées pour améliorer l'efficacité de la recherche telles que la marche aléatoire où d'autres stratégies de diversification [14].

2.2 Introduction de la Valeur *indéterminé*

Notre but ici est de fournir un cadre de travail uniforme pour représenter les affectations partielles et complètes. Concernant les affectations complètes, avec les valeurs classiques *vrai* et *faux*, les règles d'interprétation logique sont bien connues mais, avec les affectations partielles, des variables non évaluées apparaissent. Nous avons donc décidé d'introduire une troisième valeur de vérité pour les représenter.

Nous ajoutons la valeur *I* (*indéterminé*) et considérons l'ensemble des valeurs de vérité $\mathcal{T} = \{I, V, F\}$ dans notre *cadre 3-valué*. Cette nouvelle valeur induit quelques changements dans les règles d'interprétation de la logique standard. Tout d'abord, la négation de chaque valeur de vérité doit être redéfinie: $\neg V = F$, $\neg F = V$ et $\neg I = I$. Ensuite, deux approches peuvent être utilisées pour définir les autres règles: une approche optimiste et une approche pessimiste.

- L'*approche optimiste* est la plus communément utilisée. En effet, toutes les méthodes explorant un

arbre de recherche sont basées sur cette approche. Une clause est considérée indéterminée si aucun de ses littéraux n'est vrai et au moins l'un d'entre eux est indéterminé. Cette approche correspond aux règles de simplification suivantes :

$$\begin{aligned} V \vee I &\rightarrow V \\ F \vee I &\rightarrow I \\ I \vee I &\rightarrow I \end{aligned}$$

- L'approche alternative que nous proposons ici est une *approche pessimiste* qui se fonde sur le fait qu'une clause est considérée comme fausse dès qu'elle possède un littéral faux et aucun de vrai. Une clause est indéterminée si tous ses littéraux sont indéterminés. Cette approche correspond aux règles suivantes :

$$\begin{aligned} V \vee I &\rightarrow V \\ F \vee I &\rightarrow F \\ I \vee I &\rightarrow I \end{aligned}$$

Dans ce cadre, les affectations partielles n'existent plus. Nous ne travaillons maintenant qu'avec des affectations trivaluées. Les variables n'ayant pas de valeur de vérité dans la logique classique à deux valeurs prennent la valeur indéterminé.

Maintenant que le cadre de travail est clairement défini, nous pouvons étudier le comportement précis des algorithmes de recherche locale dans ce cadre avec trois valeurs de vérité.

2.3 Transitions de Recherche Locale

Nous définissons un ordre partiel \sqsupset sur l'ensemble \mathcal{T} tel que $I \sqsupset V$ et $I \sqsupset F$. Dans ce contexte, pour un problème donné avec n variables, notre espace de recherche sera l'ensemble \mathcal{T}^n . La relation d'ordre \sqsupset peut être naturellement étendue à \mathcal{T}^n : $(x_1, \dots, x_n) \sqsupset (y_1, \dots, y_n)$ si et seulement si $\exists i, x_i \sqsupset y_i$ et $\nexists j, y_j \sqsupset x_j$. Nous considérons l'ordre partiel $(\mathcal{T}^n, \sqsupset)$ avec le plus grand élément $\top = (I, \dots, I)$. Cette structure peut maintenant être utilisée pour décrire de manière précise le comportement des algorithmes de recherche locale. Nous considérons un mouvement basic de recherche locale² comme une transition $A \rightarrow A'$ où $A, A' \in \mathcal{T}^n$. Puisque nous ne considérons ici que le voisinage classique, nous imposons que les transitions $(x_1, \dots, x_n) \rightarrow (y_1, \dots, y_n)$ satisfassent $\exists i, x_i \neq y_i$ et $\forall j \neq i, x_j = y_j$. Ce voisinage correspond à une distance de Hamming égale à 1 entre deux voisins, ce qui correspond à la fonction *flip*.

2. Avec trois valeurs de vérité, le concept de flip n'est plus valide. Nous préférons utiliser le concept de mouvement, pas de recherche ou transition.

D'un côté, nous nous plaçons du point de vue de la structure des affectations. Le cadre 3-valué nous permet de manipuler à la fois les affectations partielles et les affectations complètes. Comme nous l'avons déjà mentionné, les affectations partielles peuvent être vues comme une représentation d'un ensemble de plus petites affectations (par rapport à \sqsupset). Par conséquent, la diversification consiste à considérer des ensembles plus grands d'éléments de l'espace de recherche tandis que l'intensification se recentre sur moins d'éléments. La transition correspond à une exploration de niveau quand le nombre de variables indéterminées reste le même. En matière de flip, une transition de diversification rend indéterminée une variable vraie ou fausse et une transition d'intensification rend vraie ou fausse une variable indéterminée. L'exploration de niveau correspond au flip d'une variable vraie (resp. fausse) à fausse (resp. vraie).

- **Diversification** : $A \rightarrow A'$ avec $A' \sqsupset A$
- **Intensification** : $A \rightarrow A'$ avec $A \sqsupset A'$
- **Exploration de niveau** : $A \rightarrow A'$ avec $A \not\sqsupset A'$ et $A' \not\sqsupset A$

D'un autre côté, nous devons prendre en compte l'évaluation des affectations. L'introduction de la troisième valeur de vérité nous impose de redéfinir les fonctions *eval_standard* et *choix_standard* de la section 2.1 puisque le nombre de clauses indéterminées doit être pris en compte. La nouvelle fonction d'évaluation *eval* retourne le nombre de clauses vraies mais aussi le nombre de clauses indéterminées. Dans ce contexte, une affectation est meilleure qu'une autre si elle satisfait plus de clauses. Si le nombre de clauses vraies est égal pour les deux affectations, celle générant le nombre de clauses indéterminées le plus grand est considérée comme la meilleure.

$$\begin{aligned} eval: \mathcal{S} &\rightarrow (\mathbb{N}, \mathbb{N}) \\ A &\mapsto (|\{c | sat(A, c) \wedge c \in \phi\}|, \\ &\quad |\{c | indetermine(A, c) \wedge c \in \phi\}|) \end{aligned}$$

où *indetermine*(A, c) signifie que la clause c est indéterminée pour l'affectation $A \in \mathcal{S}$. Il est évident que cette fonction est dépendante du choix de l'approche puisqu'elle retourne le nombre de clauses indéterminées.

En utilisant cette fonction, nous pouvons définir Sol_ϕ , l'ensemble de toutes les solutions pour une formule ϕ : $Sol_\phi = \{A | eval(A) = (k, 0), A \in \mathcal{T}^n\}$ avec $k = |\{c | c \in \phi\}|$.

Nous définissons l'ordre $>_{eval}$ comme l'extension lexicographique ($>, >$) de l'ordre $>$ sur les paires construites par la fonction *eval*. Nous pouvons maintenant nous situer du point de vue des transitions de par cette éva-

luation.

- **Amélioration** : $A \rightarrow A'$ avec $A' >_{eval} A$
- **Détérioration** : $A \rightarrow A'$ avec $A >_{eval} A'$
- **Préservation** : $A \rightarrow A'$ avec $eval(A') = eval(A)$

En utilisant ces concepts, un mouvement peut maintenant être caractérisé par chacun des points de vue : par exemple, un mouvement peut correspondre à une transition de diversification et d'amélioration.

Ces notions sont illustrées dans l'exemple 1. En effet, dans notre cadre 3-valué, les méthodes exactes et les méthodes approchées peuvent être toutes les deux manipulées. Les arbres de recherche utilisés par les procédures de type Davis-Putnam-Logemann-Loveland sont des ensembles de transitions d'intensification et de diversification alors que les algorithmes de recherche locale classiques effectuent des ensembles d'améliorations, de détériorations et d'exploration de niveau au niveau le plus bas (sans aucune variable indéterminée).

Exemple 1: $(a \vee b) \wedge (\neg a \vee b) \wedge (a \vee \neg b)$

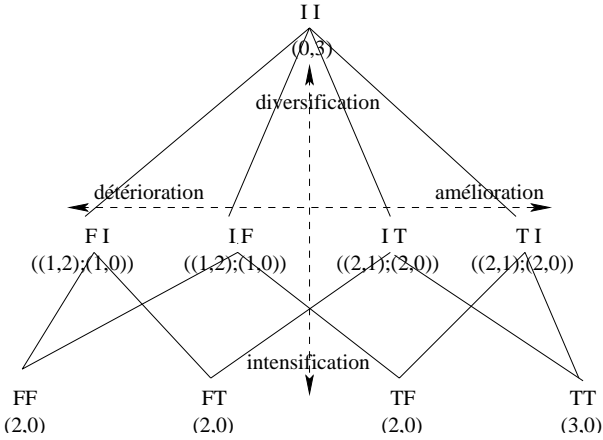


FIG. 1 – Le cadre de résolution 3-valué. Tous les éléments de \mathcal{T}^n sont représentés par rapport à la relation d'ordre \sqsupset . Une transition de diversification accroît le nombre de variables indéterminées alors qu'une transition d'intensification le diminue. Dans ce schéma, les affectations complètes sont en bas et $\top = (II)$ est en haut. Quand le nombre de variables indéterminées reste stable, les éléments sont triés avec l'ordre $>_{eval}$. Les valuations fournies par la fonction $eval$ sont données sous forme de couple $(v_1; v_2)$ où v_1 (resp. v_2) correspond à l'évaluation de l'affectation courante pour l'approche optimiste (resp. pessimiste). Ce couple est réduit à un singleton quand $v_1 = v_2$. Le meilleur élément est du côté droit et le plus mauvais du côté gauche. Par conséquent, sur cette figure, les meilleures solutions sont en bas à droite.

A ce niveau, il est important de considérer toutes les combinaisons possibles pour ces transitions issues des deux points de vue afin de construire un processus de recherche locale général.

Le but de ce processus de recherche est d'améliorer le nombre de clauses vraies. Quand ce n'est pas possible, il est préférable d'améliorer le nombre de clauses indéterminées et de réduire le nombre de clauses fausses. Nous proposons une nouvelle fonction *choix* qui sélectionne une variable dont le changement de valeur de vérité maximise le nombre de clauses vraies et minimise le nombre de clauses fausses (en augmentant le nombre de clauses indéterminées si cela est nécessaire).

Maintenant, les principaux composants pour un algorithme de recherche locale ont été fournis. Nous pouvons donc étudier leurs comportements et leurs combinaisons.

2.4 Evaluation des Règles Logiques

Nous voulons mesurer les effets des règles d'interprétation logiques présentées en partie 2.2 sur le processus de recherche. Comme décrit précédemment, un algorithme de recherche locale 3-valué (3RL) peut maintenant retourner une affectation partielle qui représente un ensemble d'affectations. Pour évaluer la qualité de ces affectations partielles, il est nécessaire d'explorer exhaustivement le sous-ensemble correspondant afin de trouver la meilleure affectation complète qui s'y trouve. Cette exploration peut être faite par un algorithme de type Branch and Bound (B&B) qui effectue une recherche exhaustive en explorant un arbre de recherche dont la racine est l'affectation partielle fournie par l'algorithme 3RL.

La Table 1 compare les effets des différentes règles logiques. Les tests ont été réalisés sur des instances 3-SAT aléatoires standards avec 500, 1000 et 2000 variables (F500.cnf, F1000.cnf et F2000.cnf). L'algorithme 3RL utilisé pour ces expériences est un algorithme Tabou 3-valué (3TS-BB) qui sera présenté en détail au chapitre suivant. 3TS-BB effectue une recherche de 5000 pas et ensuite, pour la dernière affectation partielle trouvée, le nombre de variables indéterminées ($nb. var. I$), le nombre de clauses avec une, deux et trois variables indéterminées ((I, II, III)) et la qualité ((V, I, F) après 3TS-BB) sont mémorisés. Pour estimer la qualité potentielle de l'affectation partielle trouvée par 3TS-BB, un algorithme de B&B est exécuté et fournit la meilleure affectation complète ((V, F) après B&B) en évaluant à vrai ou faux les variables indéterminées. Les résultats présentés sont une moyenne de 20 exécutions.

La Table 1 montre une claire dominance de l'interprétation pessimiste. La qualité de l'affectation partielle trouvée durant 3TS-BB est meilleure puisque le

Inst.	Observations	Optimiste	Pessimiste
500 var.	(V,I,F) après 3TS-BB	(2139,1,10)	(2145,0,5)
	nb. var. I	20	18
	(I,II,III)	(188,2,0)	(177,4,0)
	(V,F) après B&B	(2140,10)	(2145,5)
1000 var.	(V,I,F) après 3TS-BB	(4227,5,18)	(4233,0,17)
	nb. var. I	33	40
	(I,II,III)	(356,5,0)	(394,16,0)
	(V,F) après B&B	(4232,18)	(4234,16)
2000 var.	(V,I,F) après 3TS-BB	(8455,1,44)	(8465,0,35)
	nb. var. I	69	78
	(I,II,III)	(662,16,0)	(774,20,0)
	(V,F) après B&B	(8456,44)	(8465,35)

TAB. 1 – Comparaisons entre les deux approches.

nombre de clauses vraies obtenues après le B&B est supérieur au nombre obtenu par l'interprétation optimiste.

2.5 Impacte des Transitions dans le Processus de Recherche

Les interprétations optimiste et pessimiste n'induisent pas le même comportement par rapport aux différentes combinaisons de transitions caractérisant un mouvement, comme décrit dans la partie 2.3. La Table 2 présente l'évolution du nombre de clauses vraies, indéterminées et fausses (V,I,F) pour les approches optimiste (opt.) et pessimiste (pes.) et toutes les combinaisons de transitions sauf la transition "préservation" puisqu'elle n'entraîne aucun changement.

	Divers.			Intens.			Expl. de niv.		
	T	U	F	T	U	F	T	U	F
Amél. (opt.)	=	>	<	>	≤	>	>	~	~
(pes.)	=	>	<	>	≤	>	>	=	<
Dét. (opt.)	<	>	~	=	<	>	≤	~	~
(pes.)	<	>	~	=	<	>	<	=	>

TAB. 2 – Evolutions possibles du nombre de clauses vraies, indéterminées et fausses (V,I,F) provoquées par les différentes combinaisons de transitions. (>: augmente, ≥: augmente ou reste stable, =: reste stable, ≤: diminue ou reste stable, <: diminue et ~: augmente, diminue ou reste stable)

Nous pouvons remarquer des propriétés communes pour les approches optimiste et pessimiste. Quand une transition d'intensification est effectuée, il n'est jamais possible de créer des clauses indéterminées ni de supprimer des clauses vraies. Concernant la transition de diversification, aucune clause vraie ne peut être créée et aucune clause indéterminée ne peut être supprimée. Des propriétés spécifiques aux interprétations sont aussi visibles : en utilisant l'approche optimiste, aucune clause fausse ne peut être créée durant une transition de diversification tandis que pour le cas pessimiste, aucune

clause indéterminée ne peut être créée ou supprimée durant l'"exploration de niveau".

3 Cadre 3-valué utilisé dans des Algorithmes Connus

La recherche Tabou (TS)³ [6, 10] et Walksat [14] sont deux algorithmes de recherche locale bien connus. Pour voir l'intérêt du cadre de résolution 3-valué, nous avons adapté ces deux algorithmes afin d'obtenir 3TS-BB et 3Walksat-BB. Les fonctions *choix* et *eval* (chapitre 2.3) ont été utilisées afin de transformer les algorithmes en version 3-valuée.

3.1 3TS-BB

La recherche Tabou 3-valuée (3TS-BB) est une méta-heuristique de recherche locale basée sur la recherche Tabou standard. Son but est de maximiser le nombre de clauses vraies et de minimiser le nombre de clauses fausses. L'exploration de l'espace de recherche est effectuée en se déplaçant d'une affectation à une autre sélectionnée par la fonction *choix*. Ces mouvements sont guidés par la fonction *eval* qui évalue leurs bénéfices. Pour éviter les problèmes d'optima locaux, 3TS-BB utilise une liste tabou contenant des informations sur les affectations déjà visitées lors de la recherche. La taille de liste tabou est fixée à 10% du nombre de variables (valeur empirique inspirée par [10]). Nous attendons de 3-TS-BB une affectation complète pour répondre au problème MAX-SAT. Nous décidons donc de régulièrement compléter les affectations partielles avec un algorithme de B&B. Cette complétion est réalisée tous les s mouvements (empiriquement fixé à 5000). Pour réduire le temps d'exécution, B&B ne doit pas être utilisé sur des affectations partielles avec trop de variables indéterminées. Nous imposons que le nombre maximum de variables indéterminées ne doit pas être supérieur à la borne b (empiriquement fixée à 100) afin que l'effort de travail du B&B soit très faible par rapport à l'effort de 3TS-BB. La recherche s'arrête quand une solution est trouvée ou alors quand 10^6 mouvements ont été réalisés.

3.2 3Walksat-BB

Le processus général de la version 3-valuée de Walksat (3Walksat-BB) utilise les mêmes principes que 3TS-BB. Comme la version standard de Walksat, 3Walksat-BB est un algorithme de recherche locale aléatoire qui essaie de déterminer le meilleur mouvement en choisissant aléatoirement une clause indéterminée et en sélectionnant une de ces variables avec la fonction *choix*.

3. pour Tabu Search en anglais.

Nous utilisons comme base à 3Walksat-BB la version v41 de Walksat. La recherche est composée de 10 essais. Chaque essai débute avec une affectation aléatoire et après 10^5 mouvements, la meilleure affectation trouvée est complétée avec un B&B classique si c'est une affectation partielle ayant moins de b (empiriquement fixée à 100) variables indéterminées. Quand une solution est trouvée, la recherche s'arrête. Comme Walksat, 3Walksat-BB utilise l'heuristique "novelty" [14] et un bruit de 0.5 (sa valeur par défaut).

Remarque : Il est important de noter que, contrairement aux algorithmes de recherche locale classiques, les algorithmes 3-valués peuvent démarrer leur recherche avec l'affectation $\top (I, \dots, I)$. L'influence de l'affectation initiale dans les résultats est donc supprimée.

4 Résultats Expérimentaux

En raison de la nature approchée et non-déterministe de TS, 3TS-BB, Walksat et 3Walksat-BB, chaque algorithme est exécuté 20 fois sur chaque instance. Les tests sont réalisés sur un cluster avec Linux et Alinka (5 nœuds avec chacun 2 CPU Pentium IV 2.2 GHz et 1 Go de RAM) utilisé séquentiellement. Le nombre maximum de pas de recherche locale est fixé à 10^6 . Pour nos algorithmes, nous considérons qu'un retour en arrière exécuter lors du B&B à le même coût qu'un pas de recherche locale.

Toutes les instances étudiées Table 3 ont été utilisées lors de la compétition SAT 2003 [16]. Il y a des instances satisfiables et non satisfiables. Elles appartiennent à des familles d'instances différentes et aux trois classes de la compétition (industrielle, aléatoire et fait main).

Pour étudier les quatre algorithmes, nous présentons deux tables (Tables 4 and 5) : la première est une comparaison entre TS et 3TS-BB et la seconde entre Walksat et 3Walksat-BB. Deux critères de comparaison ont été utilisés : le nombre de clauses fausses (c.f.) et le nombre de mouvements (incluant les pas de recherche locale et les retours arrière du B&B) pour obtenir le meilleur résultat (mouv.). Pour ces deux critères, nous donnons la moyenne (moy.) et l'écart type (e.t.). Une autre valeur intéressante est donnée dans la dernière colonne : le pourcentage d'amélioration du nombre moyen de clauses fausses de la version 3-valorée par rapport à la version standard.

Les Tables 4 et 5 montre que le cadre de résolution 3-valoré permet d'améliorer les méthodes de recherche locale classiques telles que TS et Walksat. En effet, 3TS-BB fournit une amélioration qui peut être des fois faible et exceptionnellement négative (-1.92% pour `ezfact64_3`) mais souvent importante pour la plupart

		Familles	Instances	sat03	var.	cl.
Indust.	comb	comb1	419	5910	16804	
		comb3	421	4774	16331	
	kukula	am_6_6	362	2269	7814	
		am_7_7	363	4264	14751	
Aléatoire	des-encryption	cnf-r4-b1-k1.1-comp	416	2424	14812	
		cnf-r4-b1-k1.2-comp	417	2424	14812	
	ezfact-64	ezfact64_2	1518	3073	19785	
		ezfact64_3	1519	3073	19785	
		ezfact64_6	1522	3073	19785	
		ezfact64_7	1523	3073	19785	
	okgen	okgen-s569787048	1704	450	1935	
okgen-s1380514806		1705	450	2025		
unif	unif-r4.25-02-S1567634734	1111	500	2125		
	unif-r4.25-03-S948115330	1112	500	2125		
Fait Main	purdom	10142772393204023fw	1661	9366	37225	
		10142772393204023nc	1662	8372	33248	
		10142772393204023nw	1663	8589	34117	
	pyhala-braun	pyhala-braun-unsat-35-4-03	1543	7383	24320	
		pyhala-braun-unsat-35-4-04	1544	7383	24320	
		pyhala-braun-unsat-40-4-02	1546	9638	31795	
		pyhala-braun-unsat-40-4-03	1547	9638	31795	
	qwh	qwh.40.544	1653	2843	22558	
		qwh.40.560	1654	3100	26345	
	markstrom	mm-1x10-10-10-s.1	1488	1120	7220	
		mm-2x2-5-5-s.1	1497	324	2064	
Parity-32	par32-5-c	1539	1339	5350		
	par32-5	1540	3176	10325		

TAB. 3 – Instances sélectionnées parmi celles de la compétition SAT 2003. (sat03 → numéro de l'instance dans la compétition SAT2003, var. → variables, cl. → clauses)

des instances (+93.71% pour `par32-5`) et toutes les familles. Pour Walksat, l'amélioration est moins importante avec le cadre 3-valoré. Toutefois, 3Walksat-BB fournit des résultats intéressants car 8 familles sont améliorées (4 avec plus de 15%) et seulement 2 sont détériorées (jamais plus de -5%).

Il est intéressant de remarquer que les résultats de 3TS-BB et de 3Walksat-BB sont meilleurs alors que l'effort de recherche représenté par le nombre de mouvements est du même ordre de grandeur. Le temps d'exécution n'est pas mentionné dans les tables mais il reste à peu près le même avec et sans le cadre de résolution 3-valoré.

5 Discussion et Conclusion

Notre cadre de résolution 3-valoré nous permet de manipuler des affectations partielles et, grâce à ces affectations, des caractéristiques de la formule telles que le backbone [4] peuvent facilement être définies. Le backbone d'une formule satisfiable ϕ est l'ensemble des littéraux nécessaires. Un littéral l est nécessaire pour ϕ si et seulement si $\phi \wedge (\neg l)$ est non satisfiable. Dans le cadre 3-valoré, le backbone est l'élément $A \in Sol_\phi$ tel que $\nexists A' \sqsupset A, A' \in Sol_\phi$.

Dans cet article, nous avons proposé un nouveau cadre de résolution pour les problèmes SAT et MAX-SAT qui inclut la troisième valeur de vérité *indéterminé* dans le but de manipuler à la fois les affectations complètes mais aussi les affectations partielles. Nous avons précisément étudié l'extension et le comportement des algorithmes de recherche locale utilisant cette troisième valeur de vérité. Deux interprétations logiques peuvent être considérées et il apparaît que

	Familles	sat03	3TS-BB				TS				Amél.		
			c.f.		mouv. ($\times 10^3$)		c.f.		mouv. ($\times 10^3$)		%	c.f.	
			moy.	e.t.	moy.	e.t.	moy.	e.t.	moy.	e.t.			
Indust.	comb	419	123.90	5.79	143.10	145.53	160.20	8.70	328.83	233.63	+22.66	+25.41	
		421	79.33	5.01	243.34	190.96	110.42	4.83	156.43	87.09	+28.16		
	kukula	362	22.58	2.36	346.57	291.14	48.00	8.72	803.18	225.66	+52.96		+52.88
	363	53.33	2.93	353.74	282.04	113.00	16.20	720.32	178.27	+52.81			
Aléatoire	des-encryption	416	35.75	2.83	372.87	197.30	53.33	2.29	1.32	0.27	+32.96	+30.54	
		417	36.42	2.38	202.87	258.25	50.67	4.38	66.50	216.62	+28.12		
	ezfact-64	1518	105.00	3.37	463.88	299.16	104.58	18.72	631.44	333.80	-0.40		
		1519	102.00	4.47	388.21	209.20	100.08	11.21	669.38	315.35	-1.92		
		1522	108.83	5.26	383.12	319.05	112.33	10.92	672.35	249.31	+3.12		
		1523	110.00	4.64	488.94	287.57	113.42	15.67	623.18	276.93	+3.02		
okgen	1704	4.33	1.04	306.77	249.03	4.58	1.20	340.69	283.10	+5.46			
	1705	5.17	0.52	315.29	173.85	5.50	0.87	361.38	250.97	+6.00			
unif	1111	2.75	0.44	283.00	310.58	3.58	1.26	289.46	343.48	+23.18	+45.44		
	1112	1.75	0.60	379.46	353.53	5.42	1.43	135.37	81.25	+67.71			
Fait Main	purdom	1661	304.00	4.00	665.05	23.72	497.50	16.50	273.81	172.52	+38.89	+33.39	
		1662	256.33	9.09	462.10	300.75	350.00	3.00	401.52	259.26	+26.76		
		1663	247.50	1.50	663.28	122.31	378.00	10.00	669.85	60.97	+34.52		
	pyhala-braun	1543	239.00	3.00	92.11	20.37	377.00	0.00	269.85	30.10	+36.60		
		1544	197.00	3.74	355.73	304.16	347.00	3.00	617.31	373.33	+43.23		
		1546	342.50	10.50	25.80	4.98	464.00	0.00	571.45	0.03	+26.19		
		1547	330.00	4.00	53.95	7.58	476.00	0.00	294.21	0.04	+30.67		
	qwh	1653	125.67	9.67	52.49	63.89	130.08	10.03	33.86	60.82	+3.39		
		1654	132.00	9.05	56.67	69.93	130.08	10.84	48.06	119.94	-1.48		
	markstrom	1488	9.92	0.82	382.26	255.69	16.08	5.12	304.65	303.09	+38.31		+29.93
		1497	6.08	0.78	359.18	266.21	7.75	1.59	102.11	198.37	+21.55		
	Parity-32	1539	9.67	0.81	488.20	290.15	65.58	1.23	173.14	213.85	+85.25		+89.48
1540		9.75	1.09	509.59	225.89	155.00	1.68	192.88	262.24	+93.71			

TAB. 4 – Comparaisons entre TS et 3TS-BB.

	Instances	sat03	3Walksat-BB				Walksat				Amél.		
			c.f.		mouv. ($\times 10^3$)		c.f.		mouv. ($\times 10^3$)		%	c.f.	
			moy.	e.t.	moy.	e.t.	moy.	e.t.	moy.	e.t.			
Indust.	comb	419	69.35	5.96	548.77	303.13	70.45	7.07	569.64	273.53	+1.56	+0.33	
		421	39.60	4.09	452.94	309.93	39.25	5.62	527.25	338.57	-0.89		
	kukula	362	1.10	0.44	333.41	265.67	1.80	2.38	409.08	263.57	+38.89		+23.42
	363	71.80	14.85	519.66	292.22	78.00	14.95	555.39	304.03	+7.95			
Aléatoire	des-encryption	416	24.55	2.42	337.15	288.45	23.70	1.76	419.57	257.28	-3.59	-4.87	
		417	23.95	1.66	551.11	322.33	23.35	1.68	458.96	295.70	-2.57		
	ezfact-64	1518	70.95	3.92	544.84	279.54	86.95	4.69	393.75	280.57	+18.40		
		1519	72.70	4.56	629.37	294.40	90.40	3.57	520.31	276.70	+19.58		
		1522	73.95	4.24	518.53	284.14	88.60	2.62	308.05	267.34	+16.53		
		1523	72.55	3.93	548.05	280.23	89.80	3.67	587.71	287.36	+19.21		
okgen	1704	3.00	0.00	51.10	31.28	3.00	0.00	48.65	36.04	0.00			
	1705	4.00	0.00	147.20	140.04	4.00	0.00	191.90	168.66	0.00			
unif	1111	1.55	0.50	240.75	292.05	1.45	0.50	252.81	255.21	-6.90	-3.45		
	1112	1.00	0.00	21.61	19.49	1.00	0.00	21.20	17.01	0.00			
Fait Main	purdom	1661	372.45	12.65	635.00	270.69	412.05	12.77	487.16	248.84	+9.61	+21.61	
		1662	327.85	14.54	455.00	257.83	374.55	8.45	508.58	295.37	+12.47		
		1663	353.00	15.70	540.00	281.78	393.70	9.07	656.23	247.86	+10.34		
	pyhala-braun	1543	175.85	7.27	452.92	302.98	216.70	11.33	505.76	300.51	+18.85		
		1544	172.10	7.63	555.63	293.43	215.90	10.19	542.80	313.90	+20.29		
		1546	238.90	9.32	660.00	259.62	293.25	9.15	551.42	280.12	+18.53		
		1547	239.70	10.85	625.00	289.61	300.55	8.69	552.30	263.27	+20.25		
	qwh	1653	138.40	4.42	509.55	267.21	138.75	3.24	402.36	239.36	+0.25		
		1654	135.70	4.75	527.37	281.81	137.10	4.58	535.02	270.60	+1.02		
	markstrom	1488	10.70	1.49	570.15	239.46	11.45	1.63	400.54	269.03	+6.55		+2.75
		1497	4.85	0.57	423.88	302.64	4.80	0.51	214.27	240.11	-1.04		
	Parity-32	1539	13.80	1.89	500.11	276.64	13.85	1.68	486.16	278.46	+0.36		+0.1
1540		15.10	2.07	349.38	267.51	15.05	2.18	493.77	284.38	-0.33			

TAB. 5 – Comparaisons entre Walksat et 3Walksat-BB.

l’approche pessimiste fournit de meilleurs résultats que l’approche optimiste.

Nous avons introduit ce cadre 3-valué dans les algorithmes Tabou et Walksat et les résultats sont prometteurs. Nos travaux futurs consisteront à développer de nouvelles heuristiques spécifiques utilisant le cadre 3-valué et à étendre d’autres algorithmes à leur version

3-valuée. Ce cadre de résolution peut aussi servir de base pour de nombreuses hybridations entre des techniques de résolutions exactes et approchées.

Remerciements

Le travail présenté dans ce papier est partiellement financé dans le cadre du programme CPER COM. Nous remercions les relecteurs anonymes pour leurs commentaires très utiles.

Références

- [1] Teresa Alsinet, Felip Manyà, and Jordi Planes. Improved branch and bound algorithms for MAX-SAT. In *Proc of the 6th International Conference on the Theory and Applications of Satisfiability Testing. SAT2003*, pages 408–415, 2003.
- [2] Brian Borchers and Judith Furman. A two-phase exact algorithm for MAX-SAT and weighted MAX-SAT problems. *Journal of Combinatorial Optimization*, 2:299–306, 1999.
- [3] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, Jul 1962.
- [4] Olivier Dubois and Gilles Dequen. A backbone-search heuristic for efficient solving of hard 3-SAT formulae. In Bernhard Nebel, editor, *Proc. of the IJCAI'01*, pages 248–253, San Francisco, CA, 2001.
- [5] Michael R. Garey and David S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman & Company, San Francisco, 1979.
- [6] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, 1997.
- [7] Djamal Habet, Chu Min Li, Laure Devendeville, and Michel Vasquez. A Hybrid Approach for SAT. In *Proc. of the 8th International Conference on Constraint Programming (CP'02)*, volume 2470 of *Lecture Notes in Computer Science*, pages 172–184. Springer, 2002.
- [8] Edward A. Hirsch and Arist Kojevnikov. Unit-Walk: A new SAT solver that uses local search guided by unit clause elimination. PDMI preprint 9/2001, Steklov Institute of Mathematics at St.Petersburg, 2001.
- [9] Narendra Jussien and Olivier Lhomme. The path-repair algorithm. In *CP'99 Post-conference workshop on Large scale combinatorial optimisation and constraints*, Alexandria, VA, USA, October 1999.
- [10] Bertrand Mazure, Lakhdar Sais, and Eric Grégoire. Tabu search for SAT. In *Proc. of the AAAI-97/IAAI-97*, pages 281–285, Providence, Rhode Island, 1997.
- [11] Bertrand Mazure, Lakhdar Sais, and Eric Grégoire. Boosting complete techniques thanks to local search methods. *Annals of Mathematics and Artificial Intelligence*, 22:319–331, 1998.
- [12] Steve Prestwich. A hybrid search architecture applied to hard random 3-sat and low-autocorrelation binary sequences. In *Proc. of the 6th International Conference of Constraint Programming (CP'00)*, volume 1894, pages 337–352. Springer, 2000.
- [13] Steve Prestwich. Exploiting relaxation in local search. In *Proc. of the 1st International Workshop on Local Search Techniques in Constraint Satisfaction*, 2004.
- [14] Bart Selman, Henry A. Kautz, and Bram Cohen. Noise strategies for improving local search. In *Proc. of the AAAI'94, Vol. 1*, pages 337–343, 1994.
- [15] Bart Selman, Hector J. Levesque, and David G. Mitchell. A new method for solving hard satisfiability problems. In *Proc. of the AAAI'92*, pages 440–446, San Jose, CA, 1992.
- [16] Laurent Simon, Daniel Le Berre, and Edward A. Hirsch. The SAT2003 competition. Technical report, Sixth International Conference on the Theory and Applications of Satisfiability Testing, May 2003.
- [17] William M. Spears. Simulated annealing for hard satisfiability problems. In *Second DIMACS implementation challenge: cliques, coloring and satisfiability*, volume 26 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 533–558, 1996.
- [18] Zhao Xing and Weixiong Zhang. Efficient strategies for (weighted) maximum satisfiability. In *Proc of the 10th International Conference on Constraint Programming (CP'04)*, volume 3258 of *Lecture Notes in Computer Science*, pages 690–705. Springer, 2004.