Caractérisation de problèmes SAT à laide de la propriété de la convexité par ligne *

Hachemi Bennaceur
LIPN, Institut Galilée, Université Paris 13,
Av. J.B. Clément 93240 Villetaneuse, France,
email: hachemi.bennaceur@lipn.univ-paris13.fr

Chu Min Li
LaRIA, Université de Picardie Jules Verne,
5 Rue du Moulin Neuf, 80000 Amiens, France,
email: cli@laria.u-picardie.fr

Résumé

L'objectif de ce papier est d'étendre le lien entre le CSP et le SAT en se basant sur le codage de SAT en un CSP binaire. Sous ce codage, la consistance d'arc est équivalente à la résolution unitaire [7, 2]. Nous généralisons ce résultat en reliant la consistance de chemin avec les règles d'inférence dans le domaine de la logique propositionnelle.

Van Beek et Dechter ont identifié une propriété pour les contraintes binaires appelée "convexité par ligne" (row convexity). Cette propriété est exploitée pour identifier une classe de problèmes CSP pour lesquels la consistance de chemin est suffisante pour qu'une solution soit trouvée sans retour en arrière. Nous établissons le rapport entre la convexité par ligne et la règle de littéral pur et utilisons ce résultat pour proposer une mesure caractérisant les instances 3-SAT aléatoires satisfiables et insatisfiables. Les résultats expérimentaux nous permettent de valider cette mesure.

1 Introduction

Les liens entre les problèmes de satisfaction de contraintes (CSP) et la satisfiabilité propositionnelle (SAT) sont explorés en étudiant les correspondances entre eux [7]. Dans ce papier, nous étendons ces liens en établissant le rapport entre quelques concepts utilisés dans ces deux domaines et proposons une mesure caractérisant les instances 3-SAT aléatoires satisfiables et insatisfiables. Ce travail est basé sur le codage de SAT en un CSP binaire [1].

*Ce travail a été partiellement financé par le CNRS sous la subvention N° SUB/2001/0111/DR16

En utilisant ce codage, la structure d'un problème SAT peut être analysée en exploitant les techniques et les propriétés de la satisfaction de contraintes. C'est de cette façon qu'il a été démontré que la consistance d'arc est équivalente à la résolution unitaire [7, 2]. Dans la première partie de ce papier nous généralisons ce résultat en associant la notion de la consistance de chemin avec la résolution unitaire et binaire dans la logique propositionnelle (la résolution binaire sera définie ultérieurement). En particulier, nous montrons que la vérification de la consistance de chemin est équivalente au calcul de la fermeture de la résolution unitaire et binaire du problème SAT d'origine. Cette équivalence nous permet de définir et réaliser la consistance de chemin directement sur le problème SAT sans explicitement construire le CSP associé.

Van Beek et Dechter ont identifié une classe intéressante de CSP pour laquelle une solution peut être trouvée avec la consistance de chemin sans retour en arrière [6], chaque instance de cette classe vérifie une propriété appelée "convexité par ligne" (row convexity). Dans la seconde partie de ce papier, nous établissons le rapport entre la convexité par ligne et la règle de littéral pur et utilisons ce résultat pour proposer une mesure caractérisant les instances 3-SAT aléatoires satisfiables et insatisfiables. Les résultats expérimentaux nous permettent de valider cette mesure.

Ce papier est organisé comme suit. Dans la section 2, nous présentons quelques notions de base des problèmes SAT et CSP. La section 3 rappelle le codage de SAT en CSP et le lien entre la notion de la consistance de CSP et les règles d'inférence de la logique propositionnelle. Dans la section 4, nous rappelons d'abord la propriété de la convexité par ligne, puis nous établissons son rapport avec la règle de littéral pur avant de proposer la mesure caractérisant les instances 3-SAT aléatoires satisfiables et insatisfiables. La section 5 est consacrée aux résultats expérimentaux et la section 6 conclut.

2 Préliminaires

Nous présentons certaines notions préliminaires de SAT et le formalisme de CSP.

Une instance SAT est une conjonction de clauses. Une clause est une disjonction de littéraux. Un littéral est une variable propositionnelle x ou sa négation $\bar{x}$. Si l dénote un littéral positif ou négatif, $\bar{l}$ est son complément. Une instance SAT est satisfiable (elle a un modèle), s'il existe une affectation de ses littéraux à vrai ou faux pour que toutes ses clauses soient vraies. Si chaque clause contient k littéraux, le problème SAT est appelé k-SAT. Si exactement un littéral l apparaît négatif dans une clause et positif dans une autre, la règle de résolution consiste à générer un résolvant on l, qui est une clause contenant tous les littéraux apparaissant dans les deux clauses parents, à l'exception de l et $\bar{l}$.

Le problème de satisfaction de contraintes (CSP) consiste à trouver une affectation de valeurs aux variables pour satisfaire un ensemble de contraintes. Formellement, un CSP binaire est défini par un quadruple (X, D, C, R) où :

- X est un ensemble de n variables $\{X_1, X_2, \ldots, X_n\}$;
- D est un ensemble de n domaines $\{D_1, D_2, \ldots, D_n\}$ où chaque D_i est un ensemble de d_i valeurs possibles pour X_i ;
- C est un ensemble de m contraintes où chaque contrainte C_{ij} entre les variables X_i et X_j, $i \neq j$, est définie par sa relation R_{ij} ;
- R est un ensemble de m relations, où R_{ij} est un sous-ensemble du produit cartésien $D_i \times D_j$.
  Le prédicat $R_{ij}(v_r, v_s)$ est vrai ssi la paire (v_r, v_s) appartient à R_{ij}.
  Pour les problèmes qui nous intéressent ici, nous supposons que $(v_r, v_s) \in R_{ij} \Leftrightarrow (v_s, v_r) \in R_{ji}$.

Une solution du CSP est une affectation totale satisfaisant toutes les contraintes.

Une relation binaire entre les deux variables X_i et X_j peut être représentée par une (0-1)-matrice de $|D_i|$ lignes et $|D_j|$ colonnes en imposant un ordre sur les domaines de variables. Un zéro dans l'intersection de la ligne r et de la colonne c signifie que la paire composée du r^{ieme} élément de D_i et du c^{ieme} élément de D_j est exclue ; un 1 dans l'intersection signifie que la paire est permise.

Définition 2.1 *[6] Une ligne d'une (0-1)-matrice est convexe s'il n'y a pas deux 1 qui sont séparés par 0. Une (0-1)-matrice est convexe par ligne si toutes ses lignes sont convexes. Une relation est convexe par ligne si sa (0-1)-matrice associée est convexe par ligne.*

Dans ce papier, la notion de convexité est toujours par ligne. Nous rappelons maintenant les définitions de la consistance d'arc et de chemin.

Définition 2.2 *Un domaine D_i de D est arc consistant (2-consistant) ssi, pour chaque $v_r \in D_i$, et pour chaque $X_j \in X$ tel que $C_{ij} \in C$, il existe $v_s \in D_j$ tel que $R_{ij}(v_r, v_s)$ soit vrai. Un CSP est arc consistant ssi pour chaque $D_i \in D$, D_i est arc consistant et D_i non vide.*

Définition 2.3 *Une paire de variables X_i,X_j est chemin consistante (3-consistante) ssi pour chaque paire (v_r, v_s) tel que $R_{ij}(v_r, v_s)$ soit vrai, et pour chaque $X_k \in X$ tel que $C_{ik} \in C$ et $C_{kj} \in C$, il existe $v_t \in D_k$ tel que $R_{ik}(v_r, v_t)$ et $R_{jk}(v_s, v_t)$ soient vrais. On dira alors que v_t est le support de la paire (v_r, v_s) dans D_k. Un CSP est chemin consistant ssi chaque paire X_i,X_j est chemin consistante.*

3 La consistance de chemin et les règles d'inférence

Dans cette section nous rappelons le codage de littéral et exprimons le concept de la consistance de chemin dans la logique propositionnelle. En particulier nous montrons que la réalisation de la consistance de chemin est équivalente à l'application de la résolution unitaire et binaire.

3.1 Le codage de SAT en un CSP

Un problème SAT peut être transformé en un CSP comme suit : soit S une instance SAT avec m clauses et n littéraux, nous associons à chaque clause c_k de S une variable X_k, le domaine D_k de X_k est l'ensemble des littéraux de c_k.

Par exemple, si c_k est la clause $l_1 \vee \bar{l}_2 \vee l_3 \vee \bar{l}_4$, alors le domaine D_k de X_k est $\{l_1, \bar{l}_2, l_3, \bar{l}_4\}$.

Une contrainte entre les deux variables X_i, X_j associée aux deux clauses c_i et c_j est crée si la clause c_i contient un littéral l et la clause c_j contient son complément $\bar{l}$.

Par exemple, si c_i et c_j sont respectivement les clauses $l_1 \vee \bar{l}_2 \vee l_3$ et $\bar{l}_1 \vee l_3 \vee l_4$, une contrainte est crée entre X_i and X_j puisque c_i contient l_1 et c_j contient $\bar{l}_1$.

Pour chaque contrainte entre les deux variables X_i et X_j, une relation R_{ij} est définie par le produit cartésien D_i x D_j moins les paires $(l, \bar{l})$. Par exemple ci-dessous est la relation R_{ij} pour la contrainte ci-dessus.

$$\begin{aligned} R_{ij} \quad = \quad & \{(l_1, l_3), (l_1, l_4), (\bar{l}_2, \bar{l}_1), (\bar{l}_2, l_3), \\ & (\bar{l}_2, l_4), (l_3, \bar{l}_1), (l_3, l_3), (l_3, l_4)\} \end{aligned}$$

Pour les deux clauses c_i et c_j qui ne partagent pas de littéraux complémentaires, la relation R_{ij} est implicitement définie par le produit cartésien D_i x D_j.

Dans le reste de ce papier, S dénote un problème SAT et $P(S)$ son CSP associé. S est satisfiable ssi $P(S)$ a une solution [1].

Propiété 3.1 *[1] Si chaque clause de S contient au moins deux littéraux (resp. trois littéraux), alors $P(S)$ est arc consistant (resp. chemin consistant).*

3.2 La consistance de chemin et la résolution binaire

Définition 3.1 *Nous appelons la Résolution Unitaire (resp. Binaire) la règle de résolution appliquée aux deux clauses dont au moins une est unitaire (resp. binaire).*

Soit UR(S) la fermeture de la résolution unitaire de S. Il a été montré dans [7] que :

Théorème 3.1 *[7] La réalisation de la consistance d'arc sur $P(S)$ est équivalente au calcul de $UR(S)$.*

Le résultat suivant généralisant le théorème 3.1, établit le lien entre la consistance de chemin et la résolution unitaire et binaire.

Soit $BUR(S)$ la fermeture de la résolution unitaire et binaire de S.

Théorème 3.2 *La réalisation de la consistance de chemin dans $P(S)$ est équivalente au calcul de $BUR(S)$.*

Preuve

Nous montrons par induction qu'une paire (l_1, l_2) de R_{ij} est chemin inconsistante ssi $\bar{l}_1 \vee \bar{l}_2$ ou $\bar{l}_1$ ou $\bar{l}_2$ est une clause de S, ou une de ces clauses est produite par la résolution unitaire et binaire pendant la propagation de contraintes.

La réalisation de la consistance de chemin sur un CSP peut être faite en deux phases : la phase de vérification et la phase de propagation de contraintes. Une paire de valeurs peut être détectée inconsistante dans une de ces phases.

(i) Une paire (l_1, l_2) est détectée inconsistante pendant la phase de vérification ssi $\bar{l}_1$ ou $\bar{l}_2$ ou $\bar{l}_1 \vee \bar{l}_2$ est une clause de S, puisque (l_1, l_2) n'a pas de support dans ces clauses.

(ii) Une paire (l_1, l_2) est détectée inconsistante pendant la phase de propagation de contraintes si et seulement s'il existe une variable X_k de P(S) tel que (l_1, l_2) n'a pas de support dans D_k.

Supposons que D_k est $\{l_3, l_4, l_5\}$ (si D_k contient plus de trois éléments, le raisonnement est le même). La clause correspondant à X_k est :

$$l_3 \vee l_4 \vee l_5 \tag{1}$$

Comme la paire (l_1, l_2) de R_{ij} est chemin inconsistante, une paire dans $\{(l_1, l_3), (l_2, l_3)\}$, une paire dans $\{(l_1, l_4), (l_2, l_4)\}$, et une paire dans $\{(l_1, l_5), (l_2, l_5)\}$ sont déjà détectées chemin inconsistantes.

Par induction, une des clauses 2 et 3

$$\bar{l}_1 \vee \bar{l}_3 \tag{2}$$

$$\bar{l}_2 \vee \bar{l}_3 \tag{3}$$

et une des clauses 4 et 5

$$\bar{l}_1 \vee \bar{l}_4 \quad (4)$$
$$\bar{l}_2 \vee \bar{l}_4 \quad (5)$$

et une des clauses 6 et 7

$$\bar{l}_1 \vee \bar{l}_5 \quad (6)$$
$$\bar{l}_2 \vee \bar{l}_5 \quad (7)$$

sont générées par la résolution unitaire et binaire pendant la phase de propagation de contraintes. Supposons que les clauses 3, 4 et 7 sont les clauses générées, la résolution suivante

$$\begin{aligned} clause\ (1) \wedge clause\ (3) &\rightarrow \bar{l}_2 \vee l_4 \vee l_5 & (8)\\ clause\ (8) \wedge clause\ (4) &\rightarrow \bar{l}_1 \vee \bar{l}_2 \vee l_5 & (9)\\ clause\ (9) \wedge clause\ (7) &\rightarrow \bar{l}_1 \vee \bar{l}_2 \end{aligned}$$

mène à la clause binaire $\bar{l}_1 \vee \bar{l}_2$.
Toutes les autres possibilités mènent à une des clauses suivantes : $\bar{l}_1 \vee \bar{l}_2, \bar{l}_1, \bar{l}_2$.

■

Le théorème ci-dessus nous permet de réaliser la consistance de chemin sur un problème SAT S sans effectivement le transformer en CSP. Le calcul de $BUR(S)$ mène à un problème SAT chemin consistant (voir l'exemple 4.1).

4 Caractérisation des instances 3-SAT aléatoires satisfiables et insatisfiables

Nous établissons le rapport entre la propriété de convexité par ligne de CSP et la règle du littéral pur et nous présentons une mesure caractérisant les instances 3-SAT aléatoires satisfiables et insatisfiables.

4.1 Convexité par ligne

Van Beek et Dechter ont identifié une classe traitable de CSP pour laquelle une solution peut être trouvée en utilisant un algorithme de la consistance de chemin sans retour en arrière [6]. Le résultat est basé sur la propriété de convexité par ligne.

Définition 4.1 *[6] Étant donné un ordre des variables $X_1, ..., X_n$, un CSP binaire vérfie la convexité directionnelle par ligne si chacune de ses (0,1)-matrices, où la variable X_i apparaît avant la variable X_j dans l'ordre, est convexe par ligne.*

Théorème 4.1 *[6] Soit P un CSP binaire chemin consistant. S'il existe un ordre des variables $X_1, ..., X_n$ et des domaines $D_1, ..., D_n$ tel que P est convexe directionnelle par ligne, alors une solution de P peut être trouvée sans retour en arrière.*

Corollaire 4.1 *Le problème SAT ne contenant que de clauses binaires (2-SAT) peut être résolu en utilisant un algorithme de consistance de chemin.*

Preuve

Soit S un problème 2-SAT. Pour tout ordre de ses clauses, $P(S)$ est convexe par ligne. La réalisation de la consistance de chemin dans $P(S)$ mène soit à un problème chemin consistant pour lequel une solution peut être trouvée sans retour en arrière, soit à une relation vide, on en conclut alors que S est insatisfiable. ■

Exemple 4.1 *Soit S une instance 2-SAT définie par les clauses suivantes.*

$$\bar{l}_1 \vee \bar{l}_2 \tag{10}$$
$$\bar{l}_3 \vee l_2 \tag{11}$$
$$\bar{l}_3 \vee l_1 \tag{12}$$
$$l_3 \vee l_4 \tag{13}$$
$$l_3 \vee l_5 \tag{14}$$
$$\bar{l}_4 \vee \bar{l}_5 \tag{15}$$

Nous résolvons cette instance 2-SAT en utilisant deux méthodes équivalentes : la réalisation de la consistance de chemin dans $P(S)$ et le calcul de $BUR(S)$ (Voir le théorème 3.2).

Le CSP correspondant $P(S)$ contient 6 variables $X_1, X_2, X_3, X_4, X_5, X_6$. Les domaines de ces variables sont $D_1 = \{\bar{l}_1, \bar{l}_2\}$, $D_2 = \{\bar{l}_3, l_2\}$, $D_3 = \{\bar{l}_3, l_1\}$, $D_4 = \{l_3, l_4\}$, $D_5 = \{l_3, l_5\}$, $D_6 = \{\bar{l}_4, \bar{l}_5\}$.

La relation entre les variables X_2 et X_6 est

$$R_{26} = \{\bar{l}_3 \vee \bar{l}_4, \bar{l}_3 \vee \bar{l}_5, l_2 \vee \bar{l}_4, l_2 \vee \bar{l}_5\}$$

La phase de vérification pour $P(S)$:

$(\bar{l}_3, \bar{l}_4)$ de R_{26} n'a pas de support dans D_4,
$(\bar{l}_3, \bar{l}_5)$ de R_{26} n'a pas de support dans D_5.

Donc la paire $(\bar{l}_3, \bar{l}_4)$ et $(\bar{l}_3, \bar{l}_5)$ peut être éliminée de R_{26} ; Par conséquent, la valeur $\bar{l}_3$ peut être éliminée de D_2.

De la même façon, on déduit que la valeur $\bar{l}_3$ peut être éliminée de D_3 en utilisant la relation R_{36}.

Donc $D_2 = \{l_2\}$ et $D_3 = \{l_1\}$.

La résolution binaire et unitaire correspondant à la phase de vérification :

$$clause\ (13) \wedge clause\ (15) \quad \rightarrow \quad l_3 \vee \bar{l}_5 \tag{16}$$
$$clause\ (14) \wedge clause\ (16) \quad \rightarrow \quad l_3 \tag{17}$$
$$clause\ (11) \wedge clause\ (17) \quad \rightarrow \quad l_2 \tag{18}$$
$$clause\ (12) \wedge clause\ (17) \quad \rightarrow \quad l_1 \tag{19}$$

La phase de propagation de contraintes :

La paire (l_2, l_1) de R_{23} n'a pas de support dans D_1, qui peut être éliminée de R_{23}. D_2 et D_3 deviennent alors vides. Donc $P(S)$ n'a pas de solution.

La résolution correspondant à la phase de propagation de contraintes :

$$clause\ (10) \wedge clause\ (18) \quad \rightarrow \quad \bar{l}_1 \tag{20}$$
$$clause\ (19) \wedge clause\ (20) \quad \rightarrow \quad clause\ vide$$

Donc S est insatisfiable.

Nous avons prouvé que l'instance 2-SAT S est insatisfiable en réalisant la consistance de chemin dans $P(S)$ et en calculant $BUR(S)$. Ces deux méthodes se correspondent bien.

4.2 La convexité par ligne et la règle de littéral pur

Les résultats suivants établissent un lien entre la propriété de convexité par ligne et la règle du littéral pur.

Lemme 4.1 *Soit S une instance 3-SAT et $P(S)$ son CSP associé. Si $P(S)$ est convexe directionnelle par ligne, alors il existe dans S un littéral pur.*

Preuve

Supposons que S ne contienne pas de littéral pur. Soit c_1 une clause, pour tout ordre des littéraux de c_1, il existe une clause c_2 contenant le littéral complémenté du littéral situé au milieu de c_1. Donc, la (0-1)-matrice représentant la relation entre c_2 et c_1 n'est pas convexe par ligne. En conséquence, $P(S)$ n'est pas convexe par ligne.
■

Théorème 4.2 *Soit m le nombre de clauses de S. $P(S)$ vérifie convexité directionnelle par ligne si et seulement s'il existe une partition $S_1, ..., S_k$: $(1 \leq k \leq m)$ de l'ensemble de clauses de S tel que S contient un littéral pur qui n'apparaît que dans les clauses de S_1, et chaque ensemble de clauses $S - (S_1 \cup ... \cup S_i)$: $(1 \leq i \leq k)$ contient un littéral pur qui n'apparaît que dans les clauses de S_{i+1}.*

Preuve

$\Rightarrow$ Par construction, nous déterminons un ordre des clauses de S tel que $P(S)$ soit convexe directionnelle par ligne comme suit : toutes clauses de S_1 doivent être placées à la fin et le littéral pure de S doit être placé au milieu de chaque clause de S_1. Nous répétons la même opération pour S_2, S_3,... et S_k : les clauses de S_2 sont placées avant les clauses de S_1 et le littéral pur de $S - S_1$ est placé au milieu des clauses de S_2, de même pour $S_3, ..., S_k$. Ainsi cet ordre de clauses $S_k, S_{k-1}, ..., S_1$ mène à un CSP $P(S)$ convexe directionnelle par ligne.

$\Leftarrow$ La partition S_1,...,S_k :$(1 \leq k \leq m)$ de S est déterminée comme suit. Puisque $P(S)$ est convexe directionnelle. Selon le lemme 4.1 il existe dans S un littéral pur l_1. Soit S_1 l'ensemble de clauses dans lesquelles l_1 apparaît. Comme $P(S)$ est directionnelle convexe par ligne, $P(S - S_1)$ l'est aussi. De même, selon le lemme 4.1, il existe dans $S - S_1$ un littéral pure l_2. L'ensemble de clauses dans lesquelles l_2 apparaît forment S_2. $S_3, ..., S_k$ sont formés de la même façon. ■

4.3 La mesure caractérisant les instances aléatoires 3-SAT satisfiables et insatisfiables

Quand le CSP n'est pas convexe par ligne, nous définissons la non-convexité minimale. NC d'une instance 3-SAT par le plus petit nombre de (0,1)-matrices non convexes par ligne qui représentent les contraintes, pour tout ordre sur les littéraux et sur les clauses. Nous utilisons une heuristique simple illustrée dans la figure 1 pour estimer NC d'une instance 3-SAT S. Cette heuristique est basée sur la preuve du théorème 4.2.

Procédure MinimalRowNonConvexity(S)
Begin

```
NC :=0;
répéter
   soit neg(x) et pos(x) le nombre d'occurrences positives
   et négatives de la variable x dans S;
   Choisir x tel que neg(x)*pos(x) soit le plus petit,
   s'il y a plusieurs x dans ce cas, en choisir une tel que
   neg(x)+pos(x) soit le plus grand;
   Supprimer toutes les clauses contenant x ou x̄ dans
   S, enregistrer la date de suppression f(c) de chaque clause c;
   Trier les littéraux de chaque clause supprimée
   pour que x ou x̄ soit au milieu;
   NC :=NC+neg(x)*pos(x);
Jusqu'à ce que S devienne vide;
Écrire toutes clauses dans l'ordre décroissant de f(c);
retourner NC;
```

End

FIG. 1 – La procédure pour estimer la non-convexité minimale NC d'une instance 3-SAT S

La procédure MinimalRowNonConvexity calcule essentiellement un ordre des clauses de S initial et un ordre des littéraux dans chaque clause. Si $P(S)$ est convexe directionnelle par ligne, la procédure retourne 0 et elle est similaire à la procédure FindOrder dans [6], sinon elle calcule le nombre de (0,1)-matrices non convexes car seules les (0,1)-matrices entre les clauses supprimées en même temps peuvent avoir des 0 séparant deux 1, à cause des littéraux complémentés placés au milieu (s'il y en a).

Intuitivement, comme la procédure MinimalRowNonConvexity traite en premier toutes les clauses contenant x ou $\bar{x}$ tel que $neg(x) * pos(x)$ soit le plus petit, NC croit lentement. La suppression de ces clauses réduit le nombre d'occurrences des variables restantes, introduisant éventuellement certains littéraux pures. Pour ces raisons, nous pensons que la procédure MinimalRowNonConvexity calcule un ordre de clauses et de littéraux pour que NC soit petit.

Le calcul des occurrences des variables peut être effectué en temps $O(n+m)$, où m et n sont respectivement le nombre de clauses et de variables dans S. Donc la complexité de la procédure MinimalRowNonConvexity est $O(n(n+m))$.

5 Expérimentation

Nous utilisons la procédure MinimalRowNonConvexity pour estimer NC d'un grand nombre des instances 3-SAT. Nous varions n de 300 à 450 avec pas 50 et m pour que m/n=4, 4.1, 4.2, 4.25, 4.3, 4.4, 4.5, respectivement. Empiriquement, quand le rapport m/n est autour de 4.25, l'instance est insatisfiable avec la probabilité 0.5 (phénomène de seuil) et est la plus difficile à résoudre [5, 3].

La table 1 montre le nombre d'instances que nous avons étudié dans chaque point $(m/n, n)$.

TAB. 1 – Le nombre d'instances étudiées dans chaque point $(m/n, n)$

	300	350	400	450
4	3000	3000	3000	3000
4.1	1000	1000	600	400
4.2	1000	1000	600	400
4.25	1000	1000	600	400
4.3	1000	1000	600	400
4.4	1000	1000	600	400
4.5	1000	1000	600	400

TAB. 2 – La valeur moyenne estimée de la non convexité minimale NC des instances satisfiables dans chaque point $(m/n, n)$

	300	350	400	450
4	**1005** (2983)	**1169** (2998)	**1335** (2999)	**1500** (2999)
4.1	**1074** (944)	**1251** (972)	**1425** (582)	**1606** (392)
4.2	**1140** (715)	**1327** (746)	**1515** (452)	**1706** (307)
4.25	**1172** (551)	**1367** (547)	**1561** (319)	**1754** (224)
4.3	**1208** (370)	**1406** (353)	**1604** (197)	**1802** (128)
4.4	**1272** (128)	**1486** (85)	**1686** (46)	**1905** (27)
4.5	**1353** (32)	**1564** (11)	**1770** (5)	**2002** (4)

Pour comparer la non convexité minimale NC des instances satisfiables et insatisfiable, nous résolvons ces instances en utilisant Satz [4] et divisons les instances dans chaque point en classe satisfiable et classe insatisfiable. Les tables 2 and 3 montrent la valeur moyenne de NC (en gras) pour chaque classe et le nombre des instances dans chaque classe (entre parenthèses).

Dans le reste du papier, nous dénotons directement par NC la valeur moyenne de la non convexité minimale d'une classe d'instances. Nous avons empiriquement observé dans les tables 2 et 3 que le rapport NC/n a tendance d'être une constante pour une valeur m/n donnée et une classe, à l'exception des points où très peu d'instances are résolues. (e.g. les instances insatisfiables pour $m/n = 4$, les instances satisfiables instances pour $m/n = 4.5$). C'est particulièrement vrai pour les instances satisfiables quand $m/n \leq 4.25$ et pour les instances insatisfiables quand $m/n > 4.25$. La différence de NC/n entre les instances satisfiables et insatisfiables a tendance à devenir plus grand quand $m/n < 4.2$ et $m/n > 4.3$, ce qui pourrait être utilisé pour expliquer le phénomène de seuil. Les tables 4 et 5 montrent le rapport NC/n en gras et le nombre d'instances sur lesquelles la moyenne a été calculée entre parenthèses dans chaque point $(m/n, n)$ pour les instances satisfiables et les instances insatisfiables.

Pour vérifier le phénomène, nous résolvons un grand nombre d'instances dans chaque point

TAB. 3 – La valeur moyenne estimée de la non convexité minimale NC des instances insatisfiables dans chaque point $(m/n, n)$

	300	350	400	450
4	**1040** (17)	**1244** (2)	**1375** (1)	**1596** (1)
4.1	**1108** (56)	**1274** (28)	**1462** (18)	**1640** (8)
4.2	**1163** (285)	**1351** (254)	**1544** (148)	**1726** (93)
4.25	**1197** (449)	**1392** (453)	**1585** (281)	**1778** (176)
4.3	**1229** (630)	**1429** (647)	**1630** (403)	**1835** (272)
4.4	**1302** (872)	**1512** (915)	**1728** (554)	**1942** (373)
4.5	**1378** (968)	**1603** (989)	**1830** (595)	**2059** (396)

TAB. 4 – Rapport NC/n pour les instances satisfiables dans chaque point $(m/n, n)$

	300	350	400	450
4	**3.35** (2983)	**3.34** (2998)	**3.34** (2999)	**3.34** (2999)
4.1	**3.58** (944)	**3.57** (972)	**3.56** (582)	**3.57** (392)
4.2	**3.80** (715)	**3.79** (746)	**3.79** (452)	**3.79** (307)
4.25	**3.90** (551)	**3.91** (547)	**3.90** (319)	**3.90** (224)
4.3	**4.02** (370)	**4.02** (353)	**4.01** (197)	**4.00** (128)
4.4	**4.24** (128)	**4.25** (85)	**4.22** (46)	**4.23** (27)
4.5	**4.51** (32)	**4.49** (11)	**4.43** (5)	**4.45** (4)

TAB. 5 – Rapport NC/n pour les instances insatisfiables dans chaque point $(m/n, n)$

	300	350	400	450
4	**3.47** (17)	**3.55** (2)	**3.43** (1)	**3.55** (1)
4.1	**3.69** (56)	**3.64** (28)	**3.65** (18)	**3.64** (8)
4.2	**3.88** (285)	**3.86** (254)	**3.86** (148)	**3.84** (93)
4.25	**3.99** (449)	**3.98** (453)	**3.96** (281)	**3.95** (176)
4.3	**4.10** (630)	**4.08** (647)	**4.08** (403)	**4.08** (272)
4.4	**4.34** (872)	**4.32** (915)	**4.32** (554)	**4.32** (373)
4.5	**4.59** (968)	**4.58** (989)	**4.58** (595)	**4.58** (396)

pour $n = 220$ à $n = 380$ avec pas 20 ($n \neq 300$). 1000 instances sont résolues dans chaque point. Les tables 6 et 7 montrent le rapport NC/n dans ces points. A remarquer qu'il y a toujours très peu d'instances insatisfiables quand $m/n = 4$ et très peu d'instances satisfiables quand $m/n = 4.5$.

Il semble que le rapport NC/n pourrait être utilisé pour estimer la satisfiabilité d'une instance 3-SAT donnée. Par exemple, nous savons qu'une instance au seuil ($m/n = 4, 25$) est satisfiable avec une probabilité empirique 0,5. Cependant, si $NC/n < 3.90$, elle doit être satisfiable avec une probabilité plus grande. On pourrait imaginer qu'une des raisons de l'insatisfiabilité d'une instance est le grand nombre de ses (0-1)-matrices non convexes par ligne.

6 Conclusion

La structure d'un problème SAT est analysée à l'aide de son CSP associé avec les techniques de la satisfaction de contraintes en utilisant le codage du problème SAT en CSP. Nous avons exprimé la technique de la chemin consistance dans la logique propositionnelle et établi le rapport entre une propriété intéressante de CSP appelée convexité par ligne et la règle de littéral pur. Nous

TAB. 6 – Rapport NC/n pour les instances satisfiables dans chaque point $(m/n, n)$

	220	240	260	280	320	340	360	380
4	**3.36**	**3.36**	**3.35**	**3.35**	**3.35**	**3.34**	**3.34**	**3.34**
4.1	**3.59**	**3.58**	**3.57**	**3.58**	**3.57**	**3.56**	**3.56**	**3.56**
4.2	**3.81**	**3.81**	**3.80**	**3.80**	**3.80**	**3.79**	**3.79**	**3.79**
4.25	**3.92**	**3.91**	**3.90**	**3.91**	**3.92**	**3.89**	**3.90**	**3.90**
4.3	**4.03**	**4.02**	**4.02**	**4.02**	**4.02**	**4.00**	**4.00**	**4.01**
4.4	**4.26**	**4.23**	**4.26**	**4.25**	**4.24**	**4.22**	**4.23**	**4.24**
4.5	**4.51**	**4.47**	**4.43**	**4.48**	**4.43**	**4.44**	**4.43**	**4.51**

TAB. 7 – Rapport NC/n pour les instances insatisfiables dans chaque point $(m/n, n)$. Dans les points points (360, 4) and (380, 4), aucune instance n'est insatisfiable

	220	240	260	280	320	340	360	380
4	**3.47**	**3.47**	**3.51**	**3.46**	**3.45**	**3.53**	**-**	**-**
4.1	**3.70**	**3.69**	**3.69**	**3.65**	**3.65**	**3.66**	**3.69**	**3.61**
4.2	**3.89**	**3.89**	**3.89**	**3.88**	**3.86**	**3.86**	**3.86**	**3.85**
4.25	**4.00**	**4.00**	**3.99**	**3.99**	**3.99**	**3.97**	**3.97**	**3.96**
4.3	**4.12**	**4.11**	**4.10**	**4.09**	**4.09**	**4.07**	**4.08**	**4.07**
4.4	**4.35**	**4.35**	**4.34**	**4.34**	**4.33**	**4.31**	**4.32**	**4.31**
4.5	**4.61**	**4.60**	**4.60**	**4.59**	**4.59**	**4.57**	**4.57**	**4.57**

avons ensuite proposé une mesure basée sur cette propriété pour caractériser les instances 3-SAT satisfiables et les instances 3-SAT insatisfiables. Nous avons empiriquement observé que cette mesure tend vers une constante pour une valeur de m/n donnée. Ce résultat pourrait être utilisé pour expliquer partiellement l'insatisfiabilité d'une instance SAT.

Références

[1] Bennaceur H., *The Satisfiability problem regarded as constraint-satisfaction problem*, In the proceeding of the 12th European Conference on Artificial Intelligence, Page 125-130, Budapest, Hungary, August 1996.

[2] Bennaceur H. *A comparison between SAT and CSP techniques*, Research Report LIPN 2001-10.

[3] Crawford, J.M., Auton L.D., Experimental Results on the Crossover Point in Random 3-SAT, *Artificial Intelligence*, 81, 1996.

[4] Li C.M., Anbulagan, *Heuristics Based on Unit Propagation for Satisfiability Problems*, In Proceedings of IJCAI-97, ISBN 1-55860-480-4, Page 366-371, Nagoya, Japan, August 1997.

[5] Mitchell, D, Selman, B., Levesque H., Hard and Easy Distributions of SAT Problems, In *AAAI'92*, pages 459-465, San Jose, CA, 1992.

[6] Van Beek P., Dechter R., *On the Minimality and Global Consistency of Row-Convex Constraint Networks*, Journal of the ACM, 42 :543-561, 195.

[7] Walsh T., *SAT v CSP*, Proceedings of CP-2000, pages 441-456, Springer-Verlag LNCS-1894, 2000.