

# Un modèle markovien pour GSAT et WalkSAT résultats préliminaires

Charlotte Truchet<sup>1</sup>

Damien Noguès<sup>1,2</sup>

Narendra Jussien<sup>3,1</sup>

<sup>1</sup> LINA, UMR 6241

Université de Nantes, 2 rue de la Houssinière, 44300 Nantes, France

<sup>2</sup> Ecole Normale Supérieure

45, rue d'Ulm, 75005 Paris, France

<sup>3</sup> Ecole des Mines de Nantes

4 rue Alfred Kastler - 44307 Nantes Cedex 3 - France

charlotte.truchet@univ-nantes.fr, damien.nogues@ens.fr, narendra.jussien@emn.fr

## Résumé

Les algorithmes GSAT et WalkSAT ont un comportement bien connu expérimentalement, mais relativement peu étudié théoriquement. Nous étudions ici un modèle de GSAT et WalkSAT sous la forme de chaînes de Markov, modèle exact pour la partie gloutonne, approché pour la version avec *random restart*. Les résultats classiques sur les chaînes de Markov permettent d'en déduire deux nouvelles majorations de l'espérance du temps de calcul de WalkSAT sans *random restart*, en fonction des valeurs propres de la matrice de transition associée. Nous montrons expérimentalement sur de petites instances que cette borne permet de retrouver le paramétrage optimal observé dans la littérature. Nous donnons ensuite deux résultats sur l'espérance de GSAT ou WalkSAT avec *random restart* en fonction du nombre d'itérations avant *random restart* (entre autres). Même si les résultats restent à approfondir, ce modèle donne une piste vers une étude théorique du paramétrage optimal et au delà du comportement de ces algorithmes.

## 1 Introduction

Les algorithmes incomplets sont devenus des méthodes de choix pour la résolution de problèmes d'optimisation combinatoire en raison de leur efficacité. On s'intéressera ici aux méthodes trajectoire (par opposition aux méthodes maintenant une population de points de l'espace de recherche comme les algorithmes génétiques ou fourmis), dont les plus courants sont le tabu search, le recuit simulé ou encore les méthodes dédiées à SAT souvent dérivées de GSAT[10] et

WalkSAT[9]. Le but est d'optimiser itérativement une fonction de coût  $f$  définie sur l'espace de recherche, à partir d'un point choisi aléatoirement. On trouve toujours une étape d'optimisation gloutonne sur un voisinage du point courant, et divers mécanismes d'échappement : les méthodes gloutonnes étant par nature bloquées dans des minima locaux de  $f$ , il faut autoriser l'algorithme à détériorer  $f$  de temps en temps pour éviter ces blocages.

### 1.1 Algorithmes incomplets et probabilités

Quelle que soit la méthode de recherche locale considérée, on trouve généralement une composante aléatoire, souvent à plusieurs niveaux : choix du meilleur voisin (aléatoirement parmi les meilleurs voisins), probabilité de *random walk* pour WalkSAT, conditionnement à la température pour le recuit simulé, etc. Le comportement de ces algorithmes est donc intrinsèquement aléatoire.

La littérature fournit pléthore de tels algorithmes pour un ensemble de problèmes extrêmement vaste, ce qui donne une connaissance assez large de leur comportement expérimental. On peut considérer comme communément admis les faits expérimentaux suivants :

- (a) les algorithmes sont très efficaces (*ie* répondent en moyenne plus vite que les méthodes complètes si le problème est satisfiable) en général. Cependant, pour une même classe de problèmes (SAT par exemple), cette efficacité peut varier fortement selon les instances, même satisfiables ;

- (b) en général, le comportement de l'algorithme dépend assez fortement d'un certain nombre de paramètres : *tabu tenure*, probabilité de *random walk*, itérations avant *random restart*, etc ;
- (c) pour un problème dont la satisfiabilité est inconnue, le caractère incomplet de ces algorithmes interdit de conclure quand ils ne répondent pas. En général, la recherche est coupée après un nombre suffisamment grand d'itérations, l'information déduite étant "il est fort possible que le problème soit insoluble".

S'agissant d'algorithmes probabilistes, un certain nombre de ces observations devraient pouvoir s'expliquer au niveau théorique. Il existe pourtant assez peu de résultats théoriques, et ceux qui existent concernent principalement le recuit simulé ou les algorithmes génétiques. La convergence asymptotique est en général prouvée. C'est le cas du Tabu Search [3] ou de WalkSAT [5]. Des résultats plus précis sur l'évolution du temps de calcul en fonction du paramètre de *random walk*  $p$  sont donnés dans [7], et testés sur de petites instances.

## 1.2 Pourquoi un modèle probabiliste

Le but de ce travail est de donner un modèle probabiliste du comportement d'algorithmes de recherche locale. Qu'il s'agisse du Tabu Search, du recuit simulé ou de WalkSAT, ces algorithmes se traduisent naturellement en chaînes de Markov, objets probabilistes qui évoluent dans le temps en ne tenant pas compte du passé. Ce modèle a été largement utilisé pour prouver des convergences ou dépendances au paramétrage des algorithmes génétiques ou du recuit simulé. Sur WalkSAT, il est aussi utilisé dans [7] pour une étude du paramétrage du *random walk*, donnant des résultats statistiques vérifiés sur de petites instances.

Au niveau même du modèle, plusieurs difficultés apparaissent. D'abord, les mécanismes d'échappement comme le *random restart* ou le tabu introduisent une mémoire. Cette mémoire n'est que partielle, limitée à un certain nombre d'itérations, mais cela complique le modèle. Ensuite, la nature de l'algorithme et du problème d'optimisation peut compliquer encore le modèle, avec par exemple des voisinages complexes, des fonctions de coût délicates à exprimer ou très variables sur l'espace de recherche, etc. Pour toutes ces raisons, on s'intéresse aux algorithmes GSAT et WalkSAT, qui sont facilement exprimables sans mémoire de l'espace visité, bien connus expérimentalement, et s'attaquent à un problème propre (au sens où toutes les variables sont booléennes et ont le même rôle, et toutes les contraintes sont de la même forme).

Donner un modèle probabiliste à des algorithmes de résolution de problèmes fortement combinatoires

semble à la fois capital (il s'agit de comprendre leur comportement) et naïf : sans perte d'information, la complexité du modèle est évidemment équivalente à celle du problème de départ, ou pire. Un calcul sur le temps d'exécution par exemple, sans changer la taille du problème (nombre de variables dans le cas de SAT), est inutile car de complexité au moins égale à celle du problème de départ, difficulté que l'on rencontrera dans la suite. Dans ce cas, pourquoi donner un tel modèle ? D'abord, on peut envisager des résultats approchés ou en probabilité. Ce que dit l'intuition en (a) ou (c) est peut-être ainsi quantifiable en probabilité. Ensuite, même si les calculs pratiques sont impossibles en temps raisonnables, en théorie rien n'interdit d'obtenir des calculs exacts et de faire une approximation en pratique, ce qui permettrait d'envisager des encadrements des paramètres, cf (b). C'est le sens des résultats préliminaires donnés dans cet article.

La section 2 rappelle quelques résultats classiques sur les chaînes de Markov, et décrit le modèle de GSAT et WalkSAT dans ce formalisme. La section 3 donne deux majorations du temps de calcul pour WalkSAT en fonction de caractéristiques de la chaîne associée. Ces majorations sont d'une complexité trop grande pour être calculées systématiquement, mais on montre dans la section 4 qu'elles permettent de retrouver le paramétrage optimal observé dans la littérature pour de petites instances de SAT. Enfin, la section 5 décrit une manière d'ajouter le mécanisme de *random restart* au modèle de façon approchée.

## 2 Éléments sur les chaînes de Markov

Nous présentons ici quelques résultats classiques sur les chaînes de Markov, uniquement dans la mesure où ils sont nécessaires dans la suite. C'est un formalisme très classique en théorie des probabilités, pour une présentation complète voir par exemple [1]. Les deux premiers paragraphes sont indépendants des algorithmes considérés. Nous ne nous intéresserons qu'à des chaînes homogènes à temps discret et espace d'états fini.

### 2.1 Chaînes d'ordre 1 homogènes

Soit un espace d'états fini  $S = \{s_1 \dots s_n\}$ . Une chaîne de Markov d'ordre  $t$  sur  $S$  est une suite de variables aléatoires  $(X_i)_{i \in \mathbb{N}}$  à valeurs dans  $S$ , telle que  $X_i$  ne dépend que de  $X_{i-1} \dots X_{i-t}$ . On s'intéressera uniquement à des chaînes d'ordre 1, dont chaque variable aléatoire ne dépend que de la précédente, et homogènes, c'est-à-dire que la dépendance est constante au cours du temps.

Dans ce cas, il est pratique de représenter les probabilités de transition de  $s_i$  à  $s_j$  sous la forme d'une

matrice  $M$  appelée matrice de transition, définie par  $M_{i,j} = \mathbb{P}(X_{i+1} = s_j | X_i = s_i)$ . On vérifie aisément que  $M^p$  donne les probabilités de transition pour  $p$  itérations de la chaîne, et que si  $\mu$  est une distribution sur les états  $S$ , alors  $M^p * \mu$  est la distribution après  $p$  itérations. La matrice  $M$  peut aussi être vue comme définissant un graphe sur l'espace de recherche, dont les arêtes sont étiquetées par les probabilités de transition.

## 2.2 Chaînes absorbantes

La plupart des résultats classiques sur les chaînes de Markov concernent des chaînes irréductibles, telles que de chaque état il est possible d'atteindre tout autre état en un nombre fini d'itérations. Ce n'est hélas pas le cas des chaînes qui nous intéressent : GSAT comme WalkSAT ne ressortent pas des états solutions du problème SAT considéré. Quand l'algorithme trouve une solution, il s'arrête et la probabilité de passer à tout autre état est 0. C'est une des conditions définissant les chaînes absorbantes : intuitivement, une chaîne absorbante a des états "puits", dont elle ne peut ressortir, et de tout autre état il est possible d'atteindre un état puit.

Formellement, un état  $s_i$  d'une chaîne de Markov est dit absorbant si il est impossible à quitter, ie  $M_{i,i} = 1$  et  $M_{i,j} = 0$  pour  $i \neq j$ . Une chaîne de Markov est dite absorbante si

- elle a au moins un état absorbant,
- de chaque état, il est possible d'atteindre un état absorbant.

Dans ce cas, il est pratique d'écrire la matrice de transition en réordonnant les états : d'abord, les états non-absorbants et ensuite les états absorbants. On obtient

$$M = \left( \begin{array}{c|c} Q & R \\ \hline 0 & I \end{array} \right)$$

où  $I$  est l'identité et  $0$  est la matrice remplie de 0. On notera ainsi ces deux matrices indépendamment de leurs dimension, de même que  $M$ , pour ne pas alourdir les notations.

Intuitivement,  $R$  est la matrice qui permet de s'échapper des états transients pour arriver à une solution.  $Q$  au contraire représente le sous-graphe associé aux états transients. La chaîne boucle un certain temps dans  $Q$ , puis passe une fois par  $R$  pour tomber un état puit, et ne plus en ressortir.

Dans toute la suite on notera  $\mathbf{1} =^t (1, \dots, 1)$ ,  $Sol$  l'ensemble des états absorbants,  $\alpha$  le nombre d'états absorbants et  $T$  l'ensemble des états transients. Les matrices absorbantes vérifient les propriétés suivantes (résultats courants de la littérature, voir par exemple [1]) :

- (i) soit  $N$  l'inverse de  $I - Q$ ,  $N = I + Q + Q^2 + Q^3 \dots$   $N$  est bien définie et  $N_{i,j}$  est l'espérance du nombre de passages en  $s_j$  en partant de  $s_i$ .
- (ii) soit  $\mathbf{t} = N * \mathbf{1}$ . Soit  $\mu$  une distribution sur  $S$ . L'espérance du nombre d'itérations avant absorption, avec la distribution initiale  $\mu$ , est donnée par  $\sum_{i|s_i \in T} \mu(s_i) t_i$ .
- (iii) soit  $\|\mathbf{t}\|_1 = \sum_{1 \leq i \leq n-\alpha} |t_i|$  (norme classique sur les espaces vectoriels appelée norme un). Soit  $\mathbb{E}(\text{abs})$  l'espérance du temps d'absorption. Si la distribution de départ  $\mu$  est uniforme sur l'espace, égale à  $x$ , on a  $\mathbb{E}(\text{abs}) = x \|\mathbf{t}\|_1$ .
- (iv)  $\mathbf{t}$  est la solution de l'équation  $\mathbf{t} = Q * \mathbf{t} + \mathbf{1}$ .
- (v) la probabilité que la chaîne soit absorbée est 1, ie  $Q^z \rightarrow 0$  quand  $z \rightarrow \infty$ .

## 2.3 Expression de GSAT et WalkSAT

On traduit ici les algorithmes considérés, GSAT et WalkSAT, en chaînes de Markov. Intuitivement, ces deux algorithmes effectuent des sauts de voisinage en voisinage dans l'espace de recherche sans mémoire de l'espace visité, et la traduction doit être naturelle. Cependant, les mécanismes de *random walk* et surtout de *random restart* demandent un minimum d'attention.

Soit  $F$  une formule sur  $n$  variables,  $S$  l'espace de recherche pour l'algorithme GSAT associé (de taille  $2^n$ ),  $Sol$  l'ensemble des solutions,  $\alpha = |Sol|$  le nombre de solutions,  $f : S \rightarrow \mathbb{N}$  la fonction comptant le nombre de clauses fausses dans  $F$ ,  $V : S \rightarrow \mathcal{P}(S)$  la fonction de voisinage habituelle (*flip* d'une variable apparaissant dans une clause fausse).

On considère d'abord l'algorithme donné sur la figure 1, prenant en entrée une instanciation  $s \in S$  et un paramètre de *random walk*  $p \in [0, 1]$ . Pour  $p = 0$ , cet algorithme est la boucle gloutonne de GSAT, sans *random restart* et sans limite du nombre d'itérations. Pour  $p > 0$ , c'est la boucle interne de WalkSAT, qui ne peut plus vraiment être qualifiée de gloutonne car elle autorise des pas aléatoires sur le voisinage avec probabilité  $p$ .

Dans ce cas, l'information calculée est toujours locale à l'itération courante et strictement indépendante du passé (pas de condition d'arrêt, donc pas besoin de compter les itérations). L'espace de recherche  $S$  est l'ensemble des états noté également  $S$  ci-dessus. L'algorithme définit une chaîne de Markov d'ordre 1 dont la matrice de transition dépend de  $F$ .

Pour GSAT et WalkSAT, les états absorbants sont exactement les solutions  $Sol$  du problème SAT associé à  $F$ . On notera  $M^G$  la matrice de transition associée à GSAT,  $M^W$  celle associée à WalkSAT. Dans la suite, les notations introduites précédemment seront conservées, indicées par  $^G$  s'il s'agit de GSAT

```

1: while  $f(s) \neq 0$  do
2:    $v \leftarrow V(s)$ 
3:   if  $f(s) = 0$  then
4:     return  $s$ 
5:   else
6:     if  $\text{random}(1) \leq 1 - p$  then
7:        $s \leftarrow \text{random}(\{s' \in v, s' \text{ différent de } s \text{ d'une}$ 
          $\text{variable } x \text{ et } \textit{flipper } x \text{ optimise le nombre}$ 
          $\text{de clauses fausses}\})$ 
8:     else
9:        $s \leftarrow \text{random}(v)$ 
10:    end if
11:  end if
12: end while

```

FIG. 1 – La boucle interne de GSAT et WalkSAT, sans *random restart*, sans limite du nombre d'itérations.

et par  $W$  s'il s'agit de WalkSAT. On peut remarquer qu'en posant  $V$  la sous-matrice de taille  $2^n - \alpha$  du graphe des voisinages (tels que définis ci-dessus), on a  $Q^W = (1 - p) * Q^G + p * V$ . Enfin, la matrice  $M^G$  a une forme très particulière : sur une ligne  $i$ , si  $\beta_i < n$  est la taille du voisinage restreint (cf ligne 7 de l'algorithme donné sur la figure 1), elle vaut 0 sauf en  $\beta_i$  cases où elle vaut  $1/\beta_i$ .

## 2.4 Expression du random restart

Qu'il s'agisse de GSAT ou WalkSAT, un mécanisme de *random restart* est ajouté à la boucle donnée en 1. Toutes les  $m$  itérations, l'algorithme repart aléatoirement avec une probabilité uniforme sur  $S$ . La figure 2 rappelle précisément l'algorithme, prenant en entrée le paramètre  $p$ , et un nombre maximum d'itérations avant *random restart*  $m$ .

```

1: while  $f(s) \neq 0$  do
2:   int  $i=0$ ;
3:   while  $f(s) \neq 0$  et  $i < m$  do
4:      $s \leftarrow \text{random}(S)$ 
5:     effectuer les lignes 2 à 12 de l'algorithme figure 1
6:      $i++$ 
7:   end while
8: end while

```

FIG. 2 – Le mécanisme de *random restart*, sans limite du nombre de *random restarts*

Du point de vue des chaînes de Markov, cela signifie que l'information calculée à l'itération  $i$  ne dépend plus seulement de l'itération précédente. On a donc naturellement une chaîne d'ordre  $m + 1$ . En théorie, cela ne gêne en rien l'application du formalisme :

tout chaîne d'ordre  $t > 1$  peut en effet s'exprimer comme une chaîne d'ordre 1 équivalente, et les résultats s'appliquent. Cependant, la chaîne ainsi formée a un nombre d'états exponentiel en le nombre initial d'états, qui est déjà  $2^n$  dans notre cas.

Pour contourner ce problème on peut aussi exprimer le *random restart* sous une forme sensiblement différente : au lieu de redémarrer toutes les  $m$  itérations, on choisit à chaque itération de redémarrer, avec une probabilité de  $x = 1/(m + 1)$ , ou de continuer le fonctionnement normal de GSAT ou WalkSAT, avec une probabilité  $m/(m + 1)$ . L'avantage est bien sûr de retrouver une chaîne d'ordre 1, bien plus facile à traiter.

Cela n'est pas rigoureusement équivalent. Mais le nombre moyen de *random restarts* est le même dans les deux cas. Les chaînes qui nous intéressent ne sont pas complètement indépendantes, cependant la dépendance est limitée (ordre 1) et constante. Or pour des variables aléatoires indépendantes, les deux modèles sont équivalents. En revanche, pour des variables aléatoires pas indépendantes, on trouve aisément des cas de non-équivalence : dans le premier modèle, tout ce qui se passe après  $m$  est de l'information perdue et on peut placer des pics où l'on veut après  $m$  pour perturber l'espérance. Mais la construction suppose une forme de dépendance à long terme (supérieure à 1). L'approximation réalisée semble donc raisonnable.

## 2.5 Convergence

Le même type de modèle probabiliste est utilisé par [5] pour étudier la convergence de GSAT et WalkSAT, résultats que l'on rappelle brièvement ici. La question de la convergence de GSAT est triviale : GSAT peut rester bloqué autour d'un minimum local et ne pas terminer.

Il est également impossible de démontrer la convergence de WalkSAT : l'algorithme peut rester bloqué dans un cycle en modifiant indéfiniment le même *bit* par exemple (même si c'est extrêmement peu probable). Hoos [5] prouve que WalkSAT vérifie une propriété plus faible appelée Probabilistic Approximate Completeness (PAC). Si on note  $p_l$  la probabilité que l'algorithme (sans *restart*) ait trouvé une solution après  $l$  itérations, la propriété s'écrit

$$\lim_{l \rightarrow \infty} p_l = 1$$

**Remarque** Plus généralement, en définissant un algorithme de recherche locale comme d'habitude (itérations de voisinage en voisinage), on peut montrer qu'il y a équivalence entre "l'algorithme vérifie la PAC" et "la chaîne associée est absorbante" [8].

### 3 Majoration du temps de calcul

On cherche ici une propriété sur le temps d'absorption de la chaîne associée à WalkSAT, qui correspond au temps de calcul moyen de l'algorithme. D'après (iii), cette espérance dépend du vecteur  $\mathbf{t}$ , qui d'après (ii) mesure le temps d'absorption pour chacun des états de départ. Pour WalkSAT, la loi de départ est uniforme sur  $S$  et  $\mu$  est constante égale à  $1/2^n$ . L'espérance du temps d'absorption est donc égale à  $1/2^n \|\mathbf{t}\|_1$ . Elle dépend de la forme de  $Q$ , qui dépend elle-même de  $F$  (donc de  $n$ ). On la notera  $\mathbb{E}(F)$ .

On s'intéresse ici à  $\|\mathbf{t}\|_1$ , que l'on calcule en fonction des valeurs propres de  $Q^W$ . Dans cette section, tous les calculs concernent WalkSAT et on omettra l'indice  $W$  par souci de clarté.

Intuitivement, l'idée du calcul est la suivante : on utilise l'équation (iv) pour calculer les composantes de  $\mathbf{t}$ . Cependant, le calcul de  $Q * \mathbf{t}$  contient beaucoup trop de termes. On ne sait pas si  $Q$  est diagonalisable dans  $\mathbb{R}$  ou  $\mathbb{C}$ , mais elle est en tous cas jordanisable dans  $\mathbb{C}$ , et cela simplifie beaucoup la forme de  $Q * \mathbf{t}$ . Inconvénient, il n'existe pas nécessairement de base de Jordan orthonormée, d'où une difficulté pour prendre les normes un en fonction des coordonnées de  $\mathbf{t}$  et de  $\mathbf{1}$ . On calcule donc d'abord les composantes de  $\mathbf{t}$  dans une base de Jordan normée pour  $Q$ , en fonction des valeurs propres de  $Q$ . Ensuite, on utilise ce calcul et les propriétés de la norme un pour donner une majoration  $\|\mathbf{t}\|_1$ .

#### 3.1 Calcul de $\mathbf{t}$ dans la base de Jordan

Soit  $B = (e_k^i)_{1 \leq i \leq p, 1 \leq k \leq t_i}$  une base de Jordan pour  $Q$ , que l'on choisit normée pour la norme un, et  $Q_J$  la réduction de Jordan de  $Q$  dans cette base :

$$Q_J = \begin{pmatrix} J_1 & 0 & \dots & 0 \\ 0 & J_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & J_p \end{pmatrix}$$

où le bloc  $J_k$  de taille  $t_k$  est de la forme :

$$J_k = \begin{pmatrix} \lambda_k & 1 & & 0 \\ 0 & \ddots & \ddots & \\ \vdots & \ddots & \ddots & 1 \\ 0 & \dots & 0 & \lambda_k \end{pmatrix}$$

Soit  $\mathbf{t}_J$  l'expression de  $\mathbf{t}$  dans  $B$  et  $t_k^i$  sa coordonnée sur  $e_k^i$ . Pour calculer le produit  $Q_J * \mathbf{t}_J$  il suffit de faire le calcul dans chaque bloc de Jordan indépendamment :

$$J_i * \begin{pmatrix} t_1^i \\ \vdots \\ t_{t_i}^i \end{pmatrix} = \begin{pmatrix} \lambda_i * t_1^i + t_2^i \\ \lambda_i * t_2^i + t_3^i \\ \vdots \\ \lambda_i * t_{t_i-1}^i + t_{t_i}^i \\ \lambda_i * t_{t_i}^i \end{pmatrix}$$

L'équation (iv) se traduit de même indépendamment sur chaque bloc. On a besoin des coordonnées de  $\mathbf{1}$  dans  $B$  que l'on notera  $(u_k^i)_{i,k}$ . On a alors

$$\begin{cases} t_{t_i}^i = \lambda_i * t_{t_i}^i + u_1^i \\ t_k^i = \lambda_i * t_k^i + t_{k+1}^i + u_k^i \quad \text{si } k \neq 1 \end{cases}$$

En dépliant les équations, on en déduit une expression de  $\mathbf{t}_J$  en fonction des valeurs propres de  $Q$  :

$$t_{t_i-k}^i = \sum_{r=1}^k \frac{u_{t_i-r+1}^i}{(1-\lambda_i)^{k-r}}$$

#### 3.2 Majoration de $\|\mathbf{t}\|_1$

La base  $B$  permet de faire le calcul, mais elle n'est pas nécessairement orthonormée ce qui est gênant pour calculer  $\|\mathbf{t}\|_1$  : on a  $\mathbf{t} = \sum_{i,k} t_k^i e_k^i$ . La norme un dans la base de départ (celle qui nous intéresse), vaut donc  $\|\sum_{i,k} t_k^i e_k^i\|_1 \leq \sum |t_k^i| \|e_k^i\|_1 \leq \sum_{i,k} |t_k^i|$  par l'inégalité triangulaire et le fait que  $B$  est normée. D'où :

$$\|\mathbf{t}\|_1 \leq \sum_{i=1}^p \sum_{k=1}^{t_i} |t_k^i| = \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k \frac{|u_{t_i-r+1}^i|}{|1-\lambda_i|^{k-r}}$$

et pour tout  $i,k$ ,  $|u_k^i| \leq \|\mathbf{1}\|_1 = 2^n - \alpha$  d'où

$$\|\mathbf{t}\|_1 \leq (2^n - \alpha) * \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k \frac{1}{|1-\lambda_i|^{k-r}}$$

Et finalement

$$\mathbb{E}(F) \leq \frac{(2^n - \alpha)}{2^n} * \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k \frac{1}{|1-\lambda_i|^{k-r}}$$

On obtient une majoration de  $E(F)$ , qui représente le temps de calcul de WalkSAT, en fonction des valeurs propres de la matrice de la chaîne de Markov associée et de leur ordre de multiplicité.

#### 3.3 Majoration grossière

On donne ici une majoration grossière de  $\mathbb{E}(F)$ , en majorant d'un coup tous les termes  $\frac{1}{|1-\lambda_i|^{k-r}}$ . On utilisera la proposition 1.

**Proposition 1.** *La valeur propre  $|\lambda^*|$  de  $Q$  la plus proche de 1 (au sens où  $|1-\lambda|$  est minimal) est unique, réelle et correspond à la valeur propre de plus grand module de  $Q$ .*

*Démonstration.* La matrice  $Q$  est irréductible par construction. Par le théorème de Perron-Frobenius, le rayon spectral de  $Q$  (rayon de la plus petite boule contenant toutes les valeurs propres) est une valeur propre simple, strictement positive. Soit  $\lambda^* \in \mathbb{R}^+$  cette valeur propre. Comme toutes les valeurs propres sont dans le disque unité, cette valeur propre est l'unique valeur propre la plus proche de 1.  $\square$

On peut alors majorer  $\frac{1}{|1-\lambda_i|^{k-r}}$  par  $\frac{1}{|1-\lambda^*|^{k-r}}$  et si on pose  $t = \max_i t_i$ , par  $\frac{1}{|1-\lambda^*|^t}$ . Il s'ensuit :

$$\begin{aligned} \mathbb{E}(F) &\leq \frac{(2^n - \alpha)}{2^n} \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k \frac{1}{|1-\lambda^*|^t} \\ &\leq \frac{1}{|1-\lambda^*|^t} \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k 1 \\ &\leq \frac{1}{|1-\lambda^*|^t} \sum_{i=1}^p \sum_{k=1}^{t_i} \sum_{r=1}^k 1 \\ &\leq \frac{1}{|1-\lambda^*|^t} \sum_{i=1}^p \frac{t_i(t_i-1)}{2} \\ &\leq \frac{1}{|1-\lambda^*|^t} * p * \frac{t^2}{2} \end{aligned}$$

### 3.4 Majoration fine

La majoration donnée ci-dessus peut être améliorée en calculant plus précisément la série géométrique en  $\frac{1}{|1-\lambda_i|}$  pour tous les  $i$  tels que  $\lambda_i \neq 0$  :

$$\begin{aligned} \sum_{k=r}^{t_i} \frac{1}{|1-\lambda_i|^{k-r}} &= \frac{1 - \frac{1}{|1-\lambda_i|^{t_i-r+1}}}{1 - \frac{1}{|1-\lambda_i|}} \\ &= \frac{|1-\lambda_i|^{t_i-r+1} - 1}{|1-\lambda_i|^{t_i-r+1}} \times \frac{|1-\lambda_i|}{1 - |1-\lambda_i|} \\ &= \frac{1 - |1-\lambda_i|^{t_i-r+1}}{(1 - |1-\lambda_i|)|1-\lambda_i|^{t_i-r}} \\ &= \frac{1}{1 - |1-\lambda_i|} \left( \frac{1}{|1-\lambda_i|^{t_i-r}} - |1-\lambda_i| \right) \end{aligned}$$

D'où

$$\begin{aligned} \sum_{r=1}^{t_i} \frac{1}{|1-\lambda_i|^{t_i-r}} &= \sum_{r=0}^{t_i-1} \frac{1}{|1-\lambda_i|^r} \\ &= \frac{1 - \frac{1}{|1-\lambda_i|^{t_i}}}{1 - \frac{1}{|1-\lambda_i|}} \\ &= \frac{|1-\lambda_i|^{t_i} - 1}{|1-\lambda_i|^{t_i}} \times \frac{|1-\lambda_i|}{|1-\lambda_i| - 1} \end{aligned}$$

Et finalement

$$\begin{aligned} &\sum_{r=1}^{t_i} \sum_{k=r}^{t_i} \frac{1}{|1-\lambda_i|^{k-r}} \\ &= \frac{1}{|1-\lambda_i| - 1} \left( t_i |1-\lambda_i| + \frac{1 - |1-\lambda_i|^{t_i}}{(|1-\lambda_i| - 1)|1-\lambda_i|^{t_i-1}} \right) \end{aligned}$$

Le comportement est celui attendu : relativement petit pour les valeurs propres éloignée de 1 et augmentant très vite pour les valeurs proche de 1. Le majorant trouvé pour  $\mathbb{E}(F)$  s'exprime ainsi comme une somme de telles fonctions :

$$\begin{aligned} \mathbb{E}(F) &\leq \frac{2^n - \alpha}{2^n} \left( \sum_{i=1, \lambda_i \neq 0}^p \frac{1}{|1-\lambda_i| - 1} \right. \\ &\quad \left( t_i |1-\lambda_i| + \frac{1 - |1-\lambda_i|^{t_i}}{(|1-\lambda_i| - 1)|1-\lambda_i|^{t_i-1}} \right) \\ &\quad \left. + \sum_{i=1, \lambda_i=0}^p \sum_{k=1}^{t_i} \sum_{r=1}^k 1 \right) \end{aligned}$$

Or

$$\begin{aligned} \sum_{i=1, \lambda_i=0}^p \sum_{k=1}^{t_i} \sum_{r=1}^k 1 &= \sum_{i=1, \lambda_i=0}^p \sum_{k=1}^{t_i} k \\ &= \sum_{i=1, \lambda_i=0}^p \frac{t_i(t_i+1)}{2} \end{aligned}$$

Donc

$$\begin{aligned} \mathbb{E}(F) &\leq \frac{2^n - \alpha}{2^n} \left( \sum_{i=1, \lambda_i \neq 0}^p \frac{1}{|1-\lambda_i| - 1} \right. \\ &\quad \left( t_i |1-\lambda_i| + \frac{1 - |1-\lambda_i|^{t_i}}{(|1-\lambda_i| - 1)|1-\lambda_i|^{t_i-1}} \right) \\ &\quad \left. + \sum_{i=1, \lambda_i=0}^p \frac{t_i(t_i+1)}{2} \right) \end{aligned}$$

## 4 Tests et discussion

La section précédente donne deux majorations de l'espérance du nombre d'itérations de WalkSAT en fonction des valeurs propres de la matrice de transition associée. On vérifie d'abord expérimentalement que ces majorations sont raisonnables : elles permettent de retrouver le paramètre  $p$  optimal observé dans la littérature pour de petites instances de SAT sur lesquelles le calcul est possible. On donne ensuite quelques résultats sur les valeurs propres en question puis on discute ensuite de l'intérêt pratique de la majoration.

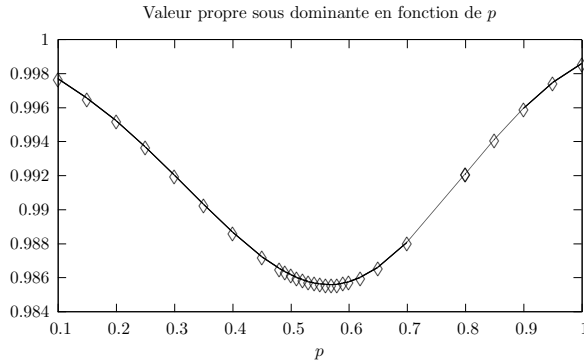


FIG. 3 – Valeur propre  $\lambda^*$  pour WalkSAT sans *random restart* en fonction de  $p$ .

#### 4.1 Vérification expérimentale

Dans l'absolu, rien ne permet de savoir si les majorations données sont proches de  $\mathbb{E}(F)$ , ou pas, en particulier pour les variations et minima.

Il est difficile de savoir théoriquement si les majorations sont fidèles ou pas. On peut cependant noter que des cas d'égalité sont possibles pour la majoration fine : les seules étapes majorantes concerne le problème de la base  $B$  qui n'est pas nécessairement orthonormée et l'expression de  $\mathbf{1}$  dans cette base. Si  $B$  est orthonormée, on a un cas d'égalité à l'étape "inégalité triangulaire" de 3.2. Même si les cas d'égalité sont difficiles à caractériser, cela constitue un indice de ce que la majoration fine n'est pas trop éloignée de la réalité. La majoration grossière ajoute une majoration de toutes les valeurs propres  $\lambda_i$  par  $\lambda^*$ , et des tailles des espaces de Jordan  $t_i$  par  $t$ , ce qui est très probablement beaucoup plus mauvais (les cas limites devenant vraiment improbables : toutes les valeurs propres égales). On la vérifie expérimentalement.

Cette majoration a été calculée expérimentalement pour mille instances aléatoires satisfiables à 20 variables, étudiées par Hoos dans [6] (disponibles dans SATLib). Les matrices associées sont de taille  $2^{20} \times 2^{20}$  et impossibles à trigonaliser entièrement en temps raisonnable avec des logiciels classiques. Cependant, elles sont incluses dans  $V$  la matrice des voisinages, au sens suivant : si  $V_{i,j} = 0$  alors  $M_{i,j} = 0$ . Elles contiennent donc au plus  $n$  termes non nul sur chaque ligne (et même en général beaucoup moins), sont des matrices creuses et des algorithmes de calcul dédiés existent. Les tests ont été faits avec la librairie SLEPc [4].

#### 4.2 Résultats

La moyenne de  $\lambda^*$  sur les mille instances de Hoos est donnée en fonction de  $p$  sur la figure 3. On observe

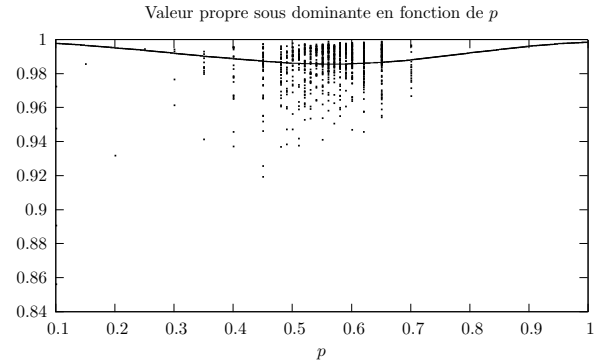


FIG. 4 – Les minima des valeurs propres sous dominantes de chaque instance en fonction de  $p$  pour les chaînes associées à WalkSAT sans *random restart*.

que l'on retrouve la valeur optimale de  $p$  observée par Hoos à 0.55. On retrouve la même observation avec le modèle de Markov de [7] avec un  $p$  optimal à 0.5. On en déduit que la majoration, au moins sur ces instances, est suffisamment fiable pour optimiser  $p$ .

Cependant, le minimum en  $p$  de la moyenne sur les mille instances ne correspond pas nécessairement à la moyenne des minima. La répartition des minima de  $\lambda^*$  en fonction de  $p$  est instructive. La figure ?? montre, pour chaque instance, la valeur minimale de  $\lambda^*$  et le  $p$  qui la réalise (en abscisse). On observe que le  $p$  réalisant le minimum de  $\lambda^*$  varie fortement selon les instances. En d'autres termes, le  $p = 0.55$  observé est un pis-aller moyen, mais certaines instances devraient être paramétrées avec un  $p$  très différent, jusqu'à 0.15 soit 85% de *random walk*.

#### 4.3 Discussion et travaux futurs

L'intérêt pratique des majorations proposées reste faible. En effet, elles sont données en fonction des valeurs propres des matrices de transition associées à WalkSAT. Or, d'une part, ces matrices sont de taille  $2^n * 2^n$ , en carré de la taille de l'espace de recherche, d'autre part, une valeur propre est une information globale sur toute la matrice : il faut connaître l'intégralité de la matrice pour la calculer.

Cependant, un certain nombre d'observations intuitives donnent l'espoir d'approximer la majoration de façon à la rendre calculable :

- les valeurs propres sont les racines du polynôme caractéristique. De manière générale, il semble bien plus probable d'obtenir un polynôme scindé qu'avec des racines multiples (cela se voit intuitivement avec la représentation en produit de mo-

nômes), de sorte que les  $t_k$  en puissance dans la majorations ne devraient pas être trop grands ;

- expérimentalement, nous avons calculé toutes les valeurs propres pour certaines instances et bien que cela soit conditionné aux approximations sur les flottants, les valeurs calculées étaient simples, ce qui confirme l'observation ci-dessus ;
- un théorème hérité de Perron-Frobenius donné par Flajolet dans [2] donne des propriétés de l'ordre de multiplicité de la valeur propre dominante en fonction des cycles du graphe associé, dont il est possible de déduire des propriétés sur les  $t_i$  ;
- enfin, les matrices associées à WalkSAT ont une forme bien particulière : sur une ligne, on a exactement  $n$  termes non nuls, et seulement deux valeurs possibles pour ces termes (selon qu'ils minimisent  $f$  ou non). C'est une information à exploiter. Ces termes dépendent du nombre de *flips* par état, dont la distribution exacte est donnée dans la matrice, et qu'on l'on doit pouvoir approcher en moyenne en temps raisonnable.

Les majorations données devraient donc pouvoir s'approximer, notamment en fonction de  $t$ .

## 5 Etude du random restart

On donne ici deux résultats préliminaires concernant le *random restart*. Ces deux résultats n'ont pas encore été testés expérimentalement, cependant, ils montrent bien le type de calculs que permet de faire le modèle probabiliste et l'intérêt qui peut en découler.

### 5.1 Random restart strict

On reprend ici le mécanisme habituel de *random restart*, celui donné sur la figure 2. Dans ce cas, il est possible de donner un encadrement de l'espérance de l'algorithme avec *random restart* en fonction de l'espérance sans *random restart* et d'un nombre raisonnable de variables ( $m$  le nombre d'itérations avant *random restart* et probabilité de répondre avant  $m$ ).

On note  $\mathbb{P}(X, s, t)$ , (resp.  $\mathbb{P}(X, s, < t)$ , etc) la probabilité pour  $X$  de passer en  $s$  pour la première fois au temps  $t$  (resp. à un temps  $t' < t$ , etc).

**Proposition 2.** Soit  $(X_i)_{i \in \mathbb{N}}$  une chaîne de Markov d'ordre 1 sur  $S$ ,  $R$  la loi uniforme sur  $S$  et  $m$  un entier. On définit la suite de variables aléatoires  $Y$  par

- $Y_i = X_{i \bmod m}$  si  $m$  ne divise pas  $i$
- $Y_i = R$  si  $m | i$

On note  $\mathbb{P}(X, s, t)$ , (resp.  $\mathbb{P}(X, s, < t)$ , etc) la probabilité pour  $X$  de passer en  $s$  pour la première fois au temps  $t$  (resp. à un temps  $t' < t$ , etc).

Soit  $s_0 \in S$ . Soit  $\mathbb{E}(Y, s_0)$  l'espérance du temps de premier passage de  $Y$  en  $s_0$ , et  $a(m)$  la probabilité que  $X$  ne passe pas en  $s_0$  avant le temps  $m$ . On a :

$$\mathbb{E}(Y, s_0) = \frac{\sum_{i=0}^{m-1} i * \mathbb{P}(X, s_0, i) + m * a(m)}{1 - a(m)}$$

*Démonstration.* On cherche d'abord  $\mathbb{P}(Y, s_0, t)$ .  $Y$  arrive pour la première fois en  $s_0$  au temps  $t$  ssi dans les  $t$  div  $m$  cycles complets effectués avant  $t$ , les itérées de  $X$  correspondantes ne sont pas passées par  $s_0$ , et le cycle commencé en  $t$  div  $m$  arrive en  $s_0$  pour la première fois en  $t \bmod m$ , ces évènements étant indépendants car  $Y$  est coupé à chaque cycle. D'où :

$$\mathbb{P}(Y, s_0, t) = \mathbb{P}(X, s_0, > m)^{t \operatorname{div} m} * \mathbb{P}(X, s_0, t \bmod m)$$

On remarque que  $\mathbb{P}(X, s_0, > m)$  ne dépend pas de  $t$ , on le note  $a$  dans la suite. L'espérance du temps de premier passage en  $s_0$  pour  $Y$  est définie par :

$$\begin{aligned} \mathbb{E}(Y, s_0) &= \sum_{t=0}^{\infty} t \mathbb{P}(Y, s_0, t) \\ &= \sum_{t=0}^{\infty} t a(m)^{t \operatorname{div} m} \mathbb{P}(X, s_0, t \bmod m) \\ &= \sum_{k=0}^{\infty} \sum_{i=0}^{m-1} (km + i) a(m)^k P(X, s_0, i) \end{aligned}$$

Comme tout converge on inverse :

$$\mathbb{E}(Y, s_0) = \sum_{i=0}^{m-1} P(X, s_0, i) \sum_{k=0}^{\infty} (km + i) a(m)^k$$

La somme intérieure est la somme de deux séries classiques en  $a$  (géométrique et dérivée de géométrique) :

$$\begin{aligned} \sum_{k=0}^{\infty} (km + i) a(m)^k &= m \sum_{k=0}^{\infty} k a(m)^k + i \sum_{k=0}^{\infty} a(m)^k \\ &= \frac{a(m) * m}{(1 - a(m))^2} + \frac{i}{1 - a(m)} \end{aligned}$$

On obtient le résultat en réinjectant dans l'expression de  $\mathbb{E}(Y, s_0)$  :

$$\begin{aligned} \mathbb{E}(Y, s_0) &= \sum_{i=0}^{m-1} P(X, s_0, i) \left( \frac{i}{1 - a(m)} + \frac{a(m) * m}{(1 - a(m))^2} \right) \\ &= \frac{1}{1 - a(m)} \sum_{i=0}^{m-1} i * P(X, s_0, i) + \frac{m * a(m)}{1 - a(m)} \end{aligned}$$

□

Le résultat est montré pour un seul état puit  $s_0$  mais la démonstration est analogue en remplaçant  $s_0$  par  $Sol$ . On vérifie notamment que pour  $m \rightarrow \infty$ ,  $\mathbb{E}(Y, s_0) \rightarrow \mathbb{E}(X, s_0)$ . On retrouve ce qui était intuitivement évident : quand le nombre d'itérations avant *random restart* tend vers l'infini, le comportement de  $Y$  devient celui de  $X$  (au moins pour l'espérance).

Ce résultat est difficile à analyser en l'état car on ne connaît pas la distribution du temps d'absorption. On peut cependant quitter le domaine rigoureux pour interpréter le résultat avec des approximations asymptotiquement raisonnables. On peut faire ici deux hypothèses :

- $a(m)$  est la probabilité de ne pas avoir trouvé de solution en  $m$  itérations, donc de rester à l'intérieur de  $T$  (ou de la matrice  $Q$ ) en  $m$  itérations, soit  $1/2^n \|Q^m\|_1 \approx 1/2^n \|Q\|_1^m$ . On pose  $\gamma = \|Q\|_1$ , et on prend  $a(m) \approx 1/2^n * \gamma^m$  ;
- plus délicat est le cas de la somme. On peut approximer  $\mathbb{P}(X, s_0, i)$  par la probabilité de ne pas être passé en  $s_0$  avant  $i$ , qui vaut  $\gamma^{i-1}$ , fois la probabilité de passer en  $s_0$  au temps  $i$ , qui vaut  $1 - \gamma$ . On a alors  $\sum_{i=0}^{m-1} i * P(X, s_0, i) \approx 1/2^n * (1 - \gamma) \sum_{i=1}^{m-1} i \gamma^{i-1}$ . C'est le début d'une série entière classique.

Les courbes correspondantes sont données sur la figure 5 pour  $\gamma = 0.99$  et  $n = 20$ . On observe que le début de la courbe (à gauche), pour des itérations de 1 à 1000, est assez fortement en dessous de  $\mathbb{E}(X, Sol)$ , ce qui justifie l'intérêt du *random restart* mais est à prendre avec précaution car l'approximation réalisée est plutôt valide asymptotiquement. Plus intéressante est la courbe asymptotique, à droite, pour des itérations de 1 à 10000 : on voit que pour des  $m$  trop grand, le *random restart* devient vite négligeable. Cela justifie la nécessité d'un paramétrage assez fin de  $m$  : à le fixer trop grand, on perd vite l'intérêt du mécanisme de *random restart*.

## 5.2 Random restart itératif

On s'intéresse maintenant au deuxième modèle, dans lequel le *random restart* consiste à effectuer un pas aléatoire avec probabilité  $1/(m+1)$ . L'étude est identique qu'il s'agisse de GSAT ou WalkSAT d'où l'omission des indices dans la suite.

On décompose comme précédemment  $M$  avec  $T$  en premier et  $Sol$  en dernier.

$$M = \left( \begin{array}{c|c} Q & R \\ \hline 0 & I \end{array} \right)$$

Soit  $M^R$  la matrice de transition de la chaîne de Markov sur  $2^n$  états, ainsi définie :

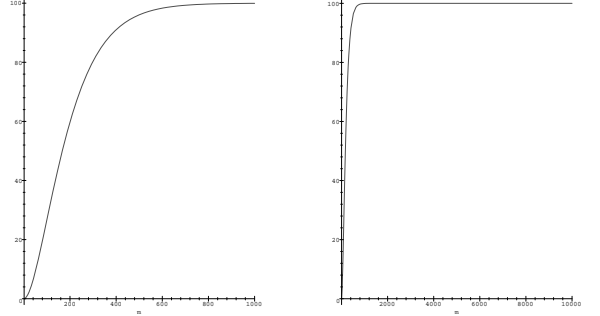


FIG. 5 – Evolution d'une approximation de l'expression de l'espérance de  $Y$  en fonction de  $m$ , le nombre d'itérations avant *random restart*.

- si  $s \in Sol$  alors la chaîne reste en  $s$  avec probabilité 1
- sinon, la chaîne saute sur tous les états  $S \setminus Sol$  avec une probabilité égale.

$M^R$  se décompose ainsi :

$$M^R = \left( \begin{array}{c|c} Q^R & R^R \\ \hline 0 & I \end{array} \right) = \left( \begin{array}{cc|cc} 1/2^n & \dots & 1/2^n & \dots \\ \vdots & & \vdots & \\ \hline & 0 & & I \end{array} \right)$$

Soit enfin  $x \in [0, 1]$ , la probabilité d'effectuer un *random restart*. On considère la chaîne de Markov d'ordre 1 dont la matrice de transition est :  $M' = x * M + (1 - x) * M^R$ . C'est une chaîne absorbante dont les états absorbants sont l'ensemble  $Sol$ . Elle se décompose en

$$M' = \left( \begin{array}{c|c} Q' & R' \\ \hline 0 & I \end{array} \right)$$

avec

$$Q' = x * Q + (1 - x) * Q^R$$

On cherche à calculer l'espérance du temps d'absorption pour la chaîne  $M'$ . Soit  $\mathbf{t}$  le vecteur des espérances du nombre de sauts avant absorption comme défini plus tôt.  $\mathbf{t}$  est solution de

$$\mathbf{t} = Q' * \mathbf{t} + \mathbf{1} = (x * Q + (1 - x) * Q^R) * \mathbf{t} + \mathbf{1}$$

Or  $Q^R$  est constante, tous termes égaux à  $1/2^n$  d'où

$$Q^R * \mathbf{t} = \frac{1}{2^n} * \sum_{1 \leq i \leq 2^n - \alpha} t_i \dots = 1/2^n \| \mathbf{t} \|_1 \mathbf{1}$$

car toutes les composantes de  $\mathbf{t}$  sont positives. On a donc

$$\mathbf{t} = xQ\mathbf{t} + ((1-x)/2^n \|\mathbf{t}\|_1 + 1) \mathbf{1}$$

En utilisant l'inégalité triangulaire et la majoration du produit des normes un, on obtient

$$\|\mathbf{t}\|_1 \leq x\|Q\|_1\|\mathbf{t}\|_1 + (1-x)/2^n \|\mathbf{t}\|_1 \|\mathbf{1}\|_1 + \|\mathbf{1}\|_1$$

Comme  $\|\mathbf{1}\|_1 = (2^n - \alpha)$ , on en déduit

$$\|\mathbf{t}\|_1 \leq \frac{2^n - \alpha}{1 - x\|Q\|_1 - (1-x)(2^n - \alpha)/2^n}$$

On obtient ainsi une majoration de l'espérance du temps de calcul de l'algorithme sans *random restart*, en fonction de l'algorithme avec. C'est une fonction décroissant très rapidement en  $x$ . C'est conforme à l'intuition :  $x = 0$  signifie un processus entièrement aléatoire sur  $S$ . Pour  $x = 1$  en revanche, la chaîne de Markov associée est seulement celle de l'algorithme sans *random restart* et l'espérance est faible. La décroissance de la courbe n'est pas très satisfaisante, car elle signifie qu'il vaut mieux ne pas faire de *random restart*, du tout. Cependant, il ne s'agit que d'une majoration. Une analyse plus poussée est nécessaire pour conclure, notamment sur la possibilité de cas d'égalité, ainsi qu'une vérification expérimentale et une comparaison aux valeurs de la littérature.

## 6 Conclusion

Cet article présente des résultats préliminaires sur l'utilisation d'un modèle de Markov pour décrire le comportement de GSAT et WalkSAT. Le modèle étant d'une complexité équivalente au problème de départ, il est inutile d'espérer faire des calculs exacts. Cependant, on montre qu'il permet de donner des encadrements ou résultats approchés sur ces deux algorithmes. L'une des majorations est confirmée expérimentalement sur de petites instances.

Il ne s'agit encore que de résultats préliminaires, dont l'application pratique n'est pas immédiate. Plusieurs sont encore à approfondir. Mais on constate déjà que le modèle permet de donner des informations notamment sur le paramétrage, qui est en général étudié statistiquement. C'est un cadre prometteur pour une approche systématique de l'étude du paramétrage et du comportement probabilistes des algorithmes de recherche locale : à terme, on peut envisager d'effectuer un paramétrage en s'appuyant sur ces résultats en moyenne ou bien en majoration.

**Remerciements** Merci à Jérémie Bourdon, LINA, UMR6241, pour ses relectures et son aide mathématique.

## Références

- [1] J. Laurie Snell Charles M. Grinstead. *Introduction to Probability : Second Revised Edition*, chapter 11. 1997.
- [2] Philippe Flajolet and Robert Sedgewick. Analytics combinatorics. preprint disponible sur <http://algo.inria.fr/flajolet/Publications/books.html>.
- [3] Fred Glover and Saïd Hanafi. Tabu search and finite convergence. *Discrete Appl. Math.*, 119(1-2) :3–36, 2002.
- [4] Vicente Hernandez, Jose E. Roman, and Vicente Vidal. Slepc : A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans. Math. Softw.*, 31(3) :351–362, 2005.
- [5] Holger H. Hoos. On the run-time behaviour of stochastic local search algorithms for SAT. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99)*, pages 661–666, Orlando, Florida, 1999.
- [6] Holger H. Hoos and Thomas Stutzle. Local search algorithms for SAT : An empirical evaluation. *Journal of Automated Reasoning*, 24(4) :421–481, 2000.
- [7] Bhaskar Krishnamachari, Xi Xie, Bart Selman, and Stephen Wicker. Analysis of random walk and random noise algorithms for satisfiability testing. *Proceedings of 6th Intl. Conference on the Principles and Practice of Constraint Programming (CP-2000)*., 1894, 2000.
- [8] Damien Noguès. Walksat et chaînes de markov. *Rapport de stage*, 2007.
- [9] Bart Selman, Henry A. Kautz, and Bram Cohen. Noise strategies for improving local search. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI'94)*, pages 337–343, Seattle, 1994.
- [10] Bart Selman, Hector J. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In Paul Rosenbloom and Peter Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, Menlo Park, California, 1992. AAAI Press.