RAIRO. RECHERCHE OPÉRATIONNELLE

A. QUILLIOT
S. GODANO

# Algorithmes de poursuite pour la résolution de programmes (linéaires) en nombres entiers

*RAIRO. Recherche opérationnelle*, tome 27, n° 3 (1993), p. 307-318

<http://www.numdam.org/item?id=RO_1993__27_3_307_0>

Recherche opérationnelle/Operations Research
(vol. 27, n° 3, 1993, p. 307 à 318)

# ALGORITHMES DE POURSUITE POUR LA RÉSOLUTION DE PROGRAMMES (LINÉAIRES) EN NOMBRES ENTIERS (*)

par A. QUILLIOT et S.GODANO ([1])

Communiqué par P. CHRÉTIENNE

Résumé. – *Nous prouvons ici que si un polyèdre P (ou un domaine suffisamment régulier) de $R^n$ contient des points entiers, alors ceux-ci peuvent presque sûrement être atteints par déplacements successifs le long des arêtes du treillis défini par une base de $Z^n$ effectués de façon à minimiser une certaine fonction distance à P, les paramètres définissant cette fonction distance étant remis à jour de façon aléatoire à chaque fois qu'un minimum local est atteint. Nous en déduisons diverses heuristiques pour la résolution de programmes à contraintes entières en discutant les stratégies possibles pour les remises à jour des paramètres ainsi que, dans le cas linéaire, les choix possibles de base pour $Z^n$.*

Mots clés : Méthodes stochastiques pour l'optimisation; programmation linéaire entière.

Abstract. – *We prove here that if some regular domain P of $R^n$ containts elements of $Z^n$, then one can almost surely reach some of these elements by moving along the edges of the lattice defined by some basis of $Z^n$ while minimizing some distance function to P, the parameters which define this random function being randomly reset every time a local optimum is attained. From this result we deduce various heuristics for dealing with programs with linear integer constraints and discuss the possible choices for a basis of $Z^n$ as well as the possible strategies for the control of the parameters which define the objective function.*

Keywords: Stochastic methods for optimization; linear integer programming.

## 1. INTRODUCTION

Il est connu (*voir* [14, 16]), que la complexité d'un problème d'optimisation combinatoire est essentiellement fonction de la forme, au sens de la convexité, du domaine défini par les contraintes. Pour une certaine notion du voisinage, la fonction économique est susceptible d'admettre un assez grand nombre d'optimum locaux. Un point courant piloté selon une stratégie de Gradient ou de Hill-Climbing risque alors de se trouver bloqué à l'intérieur d'une poche définie par un tel optimum local. Plusieurs approches ont été donc étudiées afin de, d'un point de vue pratique, contourner ce problème. On peut par

(*) Reçu en décembre 1991.
([1]) C.U.S.T., B.P. n° 206, Université Blaise-Pascal, 63174 Aubière, France.

Recherche opérationnelle/Operations Research, 0399-0559/93/03/$ 4.00

exemple s'efforcer de régulariser en l'élargissant le domaine de recherche (méthodes de pénalités, différents mode de relaxation [13, 18]). On peut aussi songer à autoriser en cours de recherche des dégradations temporaires de performances (Recuit simulé, etc. [11, 17]). On peut enfin procéder en modifiant périodiquement la fonction objective au cours du processus de recherche. Cette approche, communément dite de « poursuite » [15], car simulant un jeu de poursuite mettant au prise l'optimiseur et l'ensemble de contraintes, est utilisée par exemple pour le traitement des problèmes d'apprentissage.

C'est dans cette dernière perspective que nous nous plaçons ici. Nous commençons par considérer un polyèdre borné $P$ de $R^n$ et faire se déplacer un point courant selon une stratégie de Hill-Climbing de façon à minimiser une certaine fonction distance à $P$. La notion de voisinage est alors celle définie par la structure de treillis de $Z^n$, et les coefficients définissant la fonction distance sont remis à jour aléatoirement chaque fois qu'un minimum local est atteint. Nous prouvons qu'en général, si $P$ contient des entiers, alors ceux-ci peuvent être presque sûrement atteints par le point courant.

De ce résultat nous déduisons diverses heuristiques pour la résolution de problèmes à contraintes d'intégrité dont nous discutons les performances suivant les choix de stratégie (choix d'une base de $Z^n$ dans le cas de contraintes linéaires, mode d'ajustement des paramètres définissant la fonction objective courante, etc.) qui s'offrent au programmeur.

Il s'agit, pour ce type de problème d'une approche relativement nouvelle. Aussi bien pour l'étude des programmes linéaires entiers [3, 5, 8, 18] que pour celle des modèles généraux liés à la satisfaction des contraintes (les CSP, *voir* [10, 14]), le souci d'obtenir des solutions exactes a orienté la recherche. Ont été privilégiées les méthodes basées sur l'énumération controlée (Branch/Bound, etc.), ainsi que les techniques de coupes ou de décompositions. Cette tendance a été renforcée par le fait que pour ces problèmes très généraux, on ne dispose *a priori* pour mettre en œuvre des algorithmes d'amélioration locale que de la notion de voisinage héritée de la structure de treillis de $Z^n$, qui s'avère induire un très fort vallonnement. Quant à l'insertion du non déterminisme pour le traitement de problèmes d'Optimisation Combinatoire, elle a concerné plus les modalités de génération et d'acceptation d'une position voisine de la position courante que la flexibilité que l'on peut parfois apporter dans la définition d'un objectif de contrôle.

## 2. UN RÉSULTAT THÉORIQUE

Soit donc un polyèdre $P$ défini par des inéquations $Ax =< b$, $A$ étant une matrice $m.n$ à valeurs dans $Z$ et $b$ un vecteur dans $Z^m$, de taille de codage $M$.

Nous notons $K_k$ le $n$-cube de $Z^n$, centré à l'origine et de hauteur $k$ et posons $k\,(P) = 2n!^* 2^M$.

Pour $t$ dans $N^m$, $x$ dans $Z^n$, nous posons $D(t, x) = \sum_{i=1\ldots m} t_i.[A_i.x - b_i]^+$, avec $[u]^+$ (dénotant la partie positive de $u$) $= \mathrm{Sup}\,(u, 0)$ et $A_i$ = la ligne $i$ de la matrice $A$. Nous disons que deux éléments $x$ et $x'$ de $Z_n$ sont *voisins* si $\sup_{i=1\ldots n} |x_i - x'_i| = 1$.

Supposons maintenant $N^n$ ($N$ désigne ici l'ensemble des entiers positifs) muni d'une distribution de probabilité *n'excluant aucun élément.* Notre but étant alors de savoir si le polyèdre $P$ contient des points entiers, nous pouvons proposer l'algorithme suivant :

*Algorithme RHC* (Random Hill-Climbing)
Input : $A$, $b$ définissant un polyèdre $P$ de $R^n$ ;
Output (souhaité) : $x$ tel que $Ax \leq b$, $x$ entier ;

début
Choisir $x$ dans $Z^n$ ;
Tant que $x$ Non dans $P$ faire

début
Choisir aléatoirement $t$ dans $N^m$ ;
Possible ;
Tant que Possible faire
début
choisir aléatoirement $x'$ dans $K_{k(p)}$, voisin de $x$ et
tel que : $D\,(t, x') < D\,(t, x)$ ; (*)
Si $x'$ n'exite pas alors Non Possible sinon $x{:=}x'$ ;
fin
fin
fin.

Il vient alors :

THÉORÈME : *Dans le cas où P contient des éléments entiers et où la distribution de probabilité utilisée en (*) n'exclut aucun voisin du sommet courant, alors la probabilité qu'une exécution de l'algorithme RHC ne se termine pas est nulle.*

*Remarque* : La définition de la relation « voisin » est liée bien sûr au choix qui est fait de la base de $Z^n$. Nous supposons qu'aussi bien en cours d'exécution de l'algorithme ci-dessus qu'à l'intérieur de la démonstration qui va suivre, le choix de cette base n'est jamais remis en cause.

*Démonstration :* Il est clair que si P ne contient pas d'entier, RHC doit boucler. Réciproquement, nous savons (*voir* [12, 18]) que si $P$ contient un entier, alors cet entier peut être choisi dans l'intersection de $P$ et de $K_{k(P)}$. L'algorithme RCH fonctionne dès lors comme un cheminement dans un graphe représentatif d'une matrice de Markov. Les sommets de ce graphe correspondent au point courant de départ et aux différents minimum locaux des fonctions $D(t, x)$, $t$ dans $N^m$, $x$ dans $K_{k(P)}$. Les transitions aléatoires, autorisées correspondent aux possibles exécutions de la boucle interne de RCH. Il s'ensuit (*voir* théorie des matrices de Markov [4, 6]), que pour obtenir notre résultat, il nous suffit de prouver que si l'intersection $V$ de $P$ et de $K_{k(P)}$ est non vide, alors tout point de $K_{k(P)}$ peut être connecté à $V$ par une suite de transitions.

Soit donc $x^0$ quelconque dans $K_{k(P)}$, et $y^0$ dans V, le plus proche possible de $x^0$ pour la norme $N_0$ de $Z^n$ définie par la somme des valeurs absolue des coordonnées. Il nous suffit donc de montrer qu'il existe une transition possible à partir de $x^0$, c'est-à-dire un vecteur $t$ de $N^m$ et une succession de choix de type ($^*$) pour le déroulement de la boucle interne de RHC qui rapproche $x^0$ de $y^0$ pour la norme $N_0$ (on concluera alors par induction sur la quantité $N_0(x^0 - y^0)$ à l'existence d'une suite de transitions de $x^0$ vers $y^0$).

Nous procédons pour cela en commençant en renumérotant récursivement les contraintes définissant $P$ comme suit :

Si $x^0 \neq y^0$ (cas non trivial), il existe $i$ dans 1... $m$ tel que $A_i \,.\, x^0 < b_i$ ; On constate qu'il est alors possible de trouver $x^1$ dans $K_{k(P)}$, tel que :

$$N_0(x^0 - y^0) = N_0(x^0 - x^1) + N_0(x^1 - y^0) \; ; \; (\alpha)$$
$$A_i \,.\, x^1 \leq b_i \; ; \; (\beta)$$
$$x^1 \text{ minimise } N_0(x^0 - x^1) \qquad \text{avec} \quad (\alpha) \text{ et } (\beta) \; ;$$

On renumérote de façon à ce que $i = 1$ et on recommence le processus, en remplaçant $x^0$ par $x^1$ et en considérant uniquement les contraintes indexées de 2 à $m$.

Posons maintenant pour tout $i$ dans 1... $m$ :

$$T_i = \text{Inf} \, |[A_i x - b_i]^+ - [A_i x' - b_i]| \, ;$$

$$x, \, x' \text{ voisins dans } K_{k\,(P)}, \qquad A_i x \neq A_i x', \; A_i x > b_i \, ;$$

$$S_i = \text{Sup} \, |A_i \, . \, x - A_i \, . \, x'| \, ;$$

$$x, \, x' \text{ voisins}$$

Il suffit de choisir $t$ dans $N^m$ de telle sorte que pout tout $i$ dans $1...m$ on ait :

$$T_i . t_i > \sum_{j>i} t_j . S_i \, ;$$

pour obtenir l'existence de la transition recherchée. FIN.

*Remarque* : Il est possible, dans le résultat ci-dessus de lever l'hypothèse de linéarité des contraintes. En fait le raisonnement adopté pourra s'appliquer à des contraintes quelconques, définies par des inégalités $f_i \, (x) \leq 0$, où les $f_i$ sont des fonctions continûment différentiables, dès lors qu'il aura été possible d'établir qu'une solution entière si elle existe, peut être trouvée à l'intérieur de certaines bornes.

## 3. MISE EN ŒUVRE PRATIQUE DE L'ALGORITHME RCH : VARIANTES POSSIBLES

### 3.1. Complexité théorique et pratique de l'algorithme RCH

Supposons comme précédemment que $n$ soit la taille du vecteur inconnu, $m$ le nombre de contraintes, $k$ un nombre entier tel que la recherche de la solution au problème de contraintes considéré puisse s'effectuer dans $K_k$ et $B$ une borne supérieure imposée pour le nombre d'exécutions de la boucle externe de l'algorithme. On voit alors que chaque exécution d'une telle boucle externe impliquera au plus $2k.n$ entrées dans la boucle interne. Chaque exécution du contenu de cette boucle interne pourra en principe signifier le calcul, pour chaque voisin $x'$ de la position courante $x$, d'une quantité du type $D \, (t, \, x')$.

Nous voyons que, en l'absence de toute recherche visant à amoindrir le coût de ce calcul, la complexité-temps théorique de l'algorithme RHC s'évalue alors en $O \, (B \, . \, k \, . \, m \, . \, n^3)$, ce qui est assez élevé pour une heuristique.

Pratiquement, il s'avère qu'à l'issue de la première exécution de la boucle externe, le point courant tend à se déplacer dans un voisinage de polyèdre $P$, et que dès lors la longueur de la trajectoire du point courant entre deux remises à jour du vecteur $t$ est en général courte (en moyenne assimilable à une constante). C'est en fait au niveau de l'exécution de chaque contenu de la boucle interne que l'algorithme tend à consommer un temps important et que l'on pourra éventuellement envisager des gains.

### 3.2. Stratégie d'implémentation de l'algorithme RCH

Il est naturel de songer à extraire du résultat théorique exprimé au paragraphe précédent des algorithmes opérationnels. Cantonnons nous tout d'abord au problème de réalisabilité étudié en partie II. Les questions auxquelles nous devons répondre sont alors :

(1) Comment gérer les choix successifs sur $t$ et sur la partie (*) de la boucle interne de l'algorithme RHC ?

(2) Quand arrêter l'exécution de l'algorithme ?

(3) Quelle base de $Z^n$ choisir pour obtenir les déplacements élémentaires du point courant les plus adaptés possibles (nous ne discuterons de ce point qu'en contexte linéaire).

Nous allons discuter ces trois points successivement, en évoquant d'un point de vue général quelles sont les différentes réponses que peut suggérer l'intuition. Cette discussion sera accompagnée (paragraphe IV) d'un résumé commenté des expérimentations que nous avons pu réaliser.

DISCUSSION DE (1) : Plusieurs options sont possibles pour, à chaque entrée dans la boucle externe, générer $t$. On peut :

– procéder de façon déterministe en faisant apparaître les différents vecteurs $t$ selon un ordre ou une stratégie prédéfini. Une façon de faire peut être de :

– Choisir $t$ sur la sphère unité de $R^m$ de façon à majorer la première amélioration obtenue (difficile à faire de façon exacte) :

ou bien, en procédant de façon plus heuristique : (stratégie REGLE)

– Supposer les contraintes normalisées ;

Pour tout vecteur de base $u_i$, $i$ dans 1... $n$, déterminer celle, notée $C_i$, parmi les contraintes violées par le point courant qui est la plus améliorée par le déplacement défini par $u_i$ (ou $-u_i$) ;

Faire $t = 3$ pour toutes les contraintes $C_i$, $i$ dans 1... $n$ ;

Faire $t = 2$ pour toutes les autres contraintes violées ;

Faire $t = 1$ pour les contraintes $A_i.x \leq b_i$ qui sont satisfaites et telles que la différence $b_i - A_i.x$ soit en dessous d'une certaine valeur ;

Faire $t = 0$ pour les autres contraintes.

Cette dernière approche, qui à chaque étape écarte délibérément certaines contraintes, est bien adaptée à des problèmes contenant beaucoup de contraintes (éventuellement définies de façon implicite comme en intelligence artificielle) et facilite le calcul des quantités $D(t, x)$. On constate cependant

(*voir* partie IV) une tendance de l'algorithme à se bloquer (périodicité des vecteurs $t$) au bout de quelques exécutions de la boucle externe s'il n'est pas parvenu à faire apparaître le résultat recherché.

– Procéder par tirage au sort. Différentes lois de probabilité sont alors envisageables :

– Loi uniforme sur [0,1] ; (Stratégie UNIF)

– Loi uniforme sur $[0, 1 + [A_i.x\text{-}b_i]^+]$ pour chaque $t$, $i$ dans 1... $m$, de façon a renforcer le poids des contraintes violées ; (stratégie UNIFBIAS)

– Image de la loi normale en 0 et de variance 1 par la fonction qui à $z$ dans $R$ associe la quantité positive $e^{-z}.(1 + [A_i.x - b_i]^+)$ pour chaque $i$ dans 1... $m$ ; (stratégie IMAGE)

– Loi normale centrée en $(1 + [A_i.x - b_i]^+)$ et de variance à déterminer pour chaque $i$ dans 1... $m$. (Stratégie GAUSS)

On remarque que la dernière loi proposée autorise l'apparition dans le vecteur $t$ de coordonnées négatives. On constate en effet que si les premières exécutions de la boucle externe ont conduit le point courant dans un voisinage du domaine réel défini par les contraintes qui est relativement éloigné de toute solution entière, il devient très difficile pour le point courant de sortir de ce voisinage en collant de trop près à la frontière du domaine. On se trouve alors confronté à des mouvements de va-et-vient (que l'on peut limiter en interdisant par exemple tout mouvement directement symétrique du dernier mouvement effectué) et à des trajectoires de descente (exécution de la boucle externe) extrémement courtes. Un façon de redonner de l'espace au point courant est alors de l'autoriser à s'éloigner de certaines contraintes (on retrouve là la logique habituelle des méthodes stochastiques). Les expérimentations réalisées montrent que c'est cette dernière approche qui fournit les meilleurs résultats.

L'exécution de l'instruction (*) n'est pas non plus neutre. La validité du Théorème énoncé en partie II supposait bien qu'à l'intérieur de chaque exécution de la boucle interne, le choix d'un voisin améliorant ne soit pas déterministe. On retrouve cette exigence dans la pratique. Si l'on désire éviter d'imposer des tirages aléatoires à chaque entrée dans la boucle interne, on peut songer à modifier périodiquement l'ordre dans lequel sont visités les voisins d'un sommet quelconque (ordre dans lequel sont placés les vecteurs de la base de $Z^n$), en décidant à chaque fois de se placer sur le premier sommet améliorant que l'on rencontre.

DISCUSSION DE (2) : Nous n'avons pas de résultat théorique permettant de majorer la probabilité que le domaine considéré admette une solution entière dès lors que l'on a laissé s'effectuer un nombre donné d'itérations de l'algorithme RCH. Nous avons toutefois testé, pour un nombre d'itérations fixé assez grand (500) un certain nombre de problèmes de dimensions variables pour lesquels l'existence d'une solution entière était garantie par construction. Nous avons alors constaté que les pourcentages de réussite des différentes approches mentionnées ci-dessus évoluaient lentement en fonction des coefficients $n$ et $m$.

DISCUSSION DE (3) : (On suppose ici que les contraintes sont linéaires.) On se rend compte assez vite que les configurations d'échec pour l'algorithme RHC vont correspondre à des cas où le polyèdre défini par les contraintes est très « plat » et « allongé ». Dans ce cas, les trajectoires effectuées par le point courant lui feront traverser le polyèdre sans forcément le rapprocher des éventuelles solutions entières. On retrouve là de façon implicite la difficulté rencontrée par Lenstra [12] pour sa preuve de la polynomialité de la programmation linéaire entière en dimension finie et qui avait justifié le recours au concepts de base réduite de treillis. Afin de rendre le polyèdre de contraintes le plus régulier possible, nous avons testé (*voir* partie IV) l'artifice suivant, qui s'est avéré contribuer très sensiblement à une amélioration des résultats :

Un sommet $z$ du polyèdre ayant été calculé (le cas trivial de vacuité du polyèdre se trouve alors évacué), nous effectuons le changement de variables définie par la mise en forme normale de Hermite (*voir* [12, 18]) de la sous-matrice carrée associée à $z$. Ce faisant, nous rendons le polyèdre le plus « orthogonal » possible au voisinage de ce sommet, ce qui d'une certaine façon correspond à ce que l'on veut obtenir.

### 3.3. Variantes de l'algorithme RHC dans le cas de l'existence d'une fonction objectif

Si nous considérons maintenant un problème d'optimisation en nombres entiers donné sous la forme :

Trouver $x$ dans $Z^n$, tel que $f_i(x) \leq 0$, $i$ dans $1 \ldots m$, minimisant $f(x)$ ;

nous pouvons le traité soit par résolutions successives de problèmes de réalisabilité $f(x) \leq k$, $f_i(x) \leq 0$, $i$ dans $1 \ldots m$.

$k$ décroissant à chaque étape, soit selon une méthode de pénalité, en minimisant par exemple une quantité $g_t(x) = f(x) + \sum_{i=1 \ldots m} t_i . f_i(x)$ sur

un cube contenant le domaine de contraints, $t$ évoluant aléatoirement après chaque obtention d'un minimum local pour $g_t$, de façon à croître en norme.

Si nous considérons un problème mixte :

Trouver $x, y$ dans $Z^{n*}Rp$ tel que $f_i(x, y) \leq 0$, $i$ dans $1 \ldots m$, (domaine $P$) ; nous pouvons poser pour tout $t$ dans $N^m$ : $H_t(x, y) = \sum_{i=1 \ldots m} t_i . f_i(x, y)$ et procéder comme suit :

*Version Mixte de RCH* ;
Début
Initialiser $x, y$ dans $Z^{n*}Rp$ ;
Tant que $x, y$ Non dans $P$ faire
début
Initialiser $t = (t_1 \ldots t_m)$;
Trouver un optimum local $y'$ pour $H(x, .)$ ;
$y := y'$ ;
Tant que Non Stop faire
début
$S$ : = {voisins $z$ de $x$ tels que $H_t(x', y) < H(x, y)$};
Si $S$=Nil alors Stop sinon choisir $x'$ dans $S$ et poser $x := x'$;
fin
fin
fin.

## 4. RÉSUMÉ D'EXPÉRIMENTATIONS

Nous avons testé l'algorithme RHC pour différentes valeurs de $n$ et $m$ (les nombres de variables et de contraintes), en limitant à chaque fois à 500 le nombre de modifications sur le vecteur $t$, et en utilisant les stratégies REGLE, UNIF, UNIFBIAS, IMAGEGAUSS et GAUSS décrite en partie III pour la remise à jour de $t$. Nous nous sommes intéressés dès lors à 3 catégories d'exemples.

– Catégorie (1) : Systèmes d'inégalités et d'inéquations linéaires diophantiennes générés aléatoirement de façon à posséder au moins une solution entière dont toutes les coordonnées sont entre –10 et 10.

– Catégorie (2) : Systèmes sous la forme $Ax \leq b$, $x \geq 0$, $2cx \geq 2k - 1$, $x$ dans $Z^n$ ou $k = \text{Sup } cx$, $x \geq 0$ tels que $Ax \leq b$ et entier ; ($k$ calculé à l'aide d'une méthode exacte) ;

– Catégorie (3) : Les problèmes issus de (2) réécrits selon un changement de variables correspondant à la mise en forme normale de Hermite de la sous-matrice de contraintes définissant un des sommets du polyèdre : $x \geq 0$, $Ax \leq b$, $2\,cx \geq 2k - 1$.

Les questions que nous avons posées ont alors été les suivantes :

– L'algorithme a-t-il identifié une solution entière ?

– Au bout de combien d'exécutions de la boucle externe ?

Nous avons par ailleurs comparé nos méthodes, sur chacun des exemples testés, avec une recherche par recuit simulé, conçue en utilisant dans $Z^n$ la même notion du voisinage que l'algorithme RHC et utilisant les paramètres suivant :

– 500 tirages par descentes ;

– Probabilité d'acceptation d'un tirage défavorable $e^{-\Delta D\,(t,\,x)/T\,(k)}$ où $T\,(k)$ à la température au cours de la $k^\circ$ descente ;

– Température initialisée à 100, décroissant de 2 en 2 jusqu'à 10, puis de 1 en 1 jusqu'à 1, de 0.1 en 0.1 jusqu'à 0.1, de 0.01 jusqu'à 0.01.

Les résultats que nous avons obtenus et portant sur des valeurs de $n$ allant de 10 à 50 ainsi que des valeurs de $m$ allant de 30 à 150 ne font pas apparaître d'infléchissement significatif des performances en fonction de $n$ et $m$. Pour cette raison, les tableaux ci-dessous ne concernent que le cas $n = 50$ ($m$ variant de 70 à 150). Ces tableaux concernent tous 20 problèmes [catégories (1), (2), et (3)], testés selon les 5 stratégies mentionnées plus haut et selon le recuit simulé et contiennent dans chaque case :

– Le nombre d'échecs enregistrés ;

– La moyenne du nombre d'exécutions de la boucle externe nécessaires pour atteindre une solution (dans le cas d'un succès) ;

– Le nombre maximum d'exécutions de la boucle externe qu'il a fallu pour arriver à une solution (en cas de succès).

CATÉGORIE DE PROBLÈMES (1) : (pour 20 essais)

| | UNIF | UNIFBIAS | IMAGE | GAUSS | REGLE | RECUIT |
|---|---|---|---|---|---|---|
| Échec . . . . . | 3 | 2 | 2 | 0 | 4 | 0 |
| Moyenne . . . . . | 19 | 23 | 25 | 38 | 7 | |
| Durée-max . . . . . | 285 | 254 | 301 | 452 | 31 | |

CATÉGORIE DE PROBLÈMES (2) : (pour 20 essais)

| | UNIF | UNIFBIAS | IMAGE | GAUSS | REGLE | RECUIT |
|---|---|---|---|---|---|---|
| Échec . . . . . | 6 | 4 | 3 | 1 | 7 | 3 |
| Moyenne . . . . . | 24 | 28 | 36 | 63 | 10 | |
| Durée-max . . . . . | 326 | 352 | 465 | 484 | 38 | |

CATÉGORIE DE PROBLÈMES (3) : (pour 20 essais)

| | UNIF | UNIFBIAS | IMAGE | GAUSS | REGLE | RECUIT |
|---|---|---|---|---|---|---|
| Échec . . . . . | 3 | 2 | 2 | 1 | 4 | 1 |
| Moyenne . . . . . | 13 | 16 | 18 | 28 | 5 | |
| Durée-max . . . . . | 122 | 89 | 187 | 261 | 15 | |

## 5. CONCLUSIONS

Il est toujours difficile d'évaluer la validité d'une approche stochastique. Ici, se mélangent les considérations liées aux propriétés de la relation de voisinage considérée avec le point de vue particulier des algorithmes de poursuite. L'intérêt des méthodes que nous venons de décrire est qu'elles peuvent *a priori* s'appliquer de façon satisfaisante pour tout type de problèmes portant sur des variables entières, et ce en exploitant une relation de voisinage que l'on peut qualifier de minimale. Mais certaines difficultés doivent être contournées : il faut parvenir à se décoller périodiquement du domaine défini par les contraintes relâchées pour éviter de trouver bloqué, et il est souhaitable de parvenir à une définition de l'objectif mobile qui ne rende pas le calcul de celui-ci trop coûteux.

Le point de vue qui consiste à faire porter le nom déterminisme non pas sur le mode de génération ou d'acceptation d'une position voisine de la position courante, mais sur la définition d'un l'objectif modulable en cours d'algorithme, est cependant en soi prometteur, dès lors que l'on se limite à la recherche d'heuristiques. On constate qu'il induit une assez grande capacité de l'algorithme à éviter le piège des optimum locaux et on peut imaginer pour cette approche d'autres modalités d'application (par exemple dans le cadre de problèmes à objectifs multicritères ou dans celui de la conception de résolveurs automatiques pour problèmes de satisfaction de contraintes).

## RÉFÉRENCES

1. C. BERGER, *Graphes et hypergraphes*, Dunod, 1974.
2. R. BIXBY et W. CUNINGHAM, Converting linear Programs to Network Problems, *Maths. of Operat. Research, 5*, 1098, p. 321-357.
3. V. CHVATAL, *Linear programming*, Freeman, New York, 1983.
4. P. CHRÉTIENNE et R. FAURE, *Processus stochastiques, leurs graphes et usages*, Gauthier-Villars, 1974.
5. J. EDMONDS et F. GILES, Total dual integrality of linear inequality systems, in *Progress in Combinatorial optimization*, Academic Press, Toronto, 1984, p. 117-129.

6. S. FELLER, *An Introduction to Probability Theory and its Application*, New York-London, John Wiley, 1957.
7. M. GAREY et D. JOHNSSON, *Computer and Intractability*, W. Freeman and Co, New York, 1979.
8. F. GILES et W. PULLEYBLANK, Total dual integrality and integer polyedra, *Linear Algebra and Application, 25*, 1979, p. 191-196.
9. A. HOFFMAN et J. KRUSTAL, Integral boundary points of convex polyedra, in *Linear inequalities and related systems*, H. Kuhn and A. Tucker eds., Princeton Univ. Press, 1986, p. 223-246.
10. L. KANAL et V. KUMAL, *Search in Artificial Intelligence*, Springer-Verlag, 1988.
11. S. KIRKPATRICK, C. GELATT et M. VECCHIT, *Optimization by simulated annealing RC 9355*, 4/2/82 Computer Science/Engineering technology, IBM T. J. Watson Center New York.
12. H. LENSTRA, Integer programming with a fixed number of variables, *Maths. of Operat. Research, 8*, 1983, p. 538-548.
13. M. MINOUX, *Programmation Mathématiques : Théorie et Algorithmes*, Tomes 1, 2, Dunod, 1983.
14. C. PAPADIMITRIOU et K. STEIGLITZ, *Combinatorial optimization* (chap. 3, 4, 5), Prentice Hall, 1982.
15. J. PEARL, *Heuristics*, Prentice Hall, 1986.
16. M. SAKAROVICTH, *Optimization combinatoire*, Herman, 1984.
17. P. SIARRY et G. DREYFUS, *La méthode du recuit simulé*, I.D.E.S.T., Paris, 1988.
18. A. SCHRIJVER, *Theory of linear and integer programming*, (chap. 19, 20), John Wiley Interscience, 1986.