CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
(Université Paris-Dauphine)
Unité de Recherche Associée au CNRS n° 825

UNE APPROCHE HYBRIDE DE RÉSOLUTION DE PROBLÈMES LINÉAIRES GÉNÉRAUX EN NOMBRES ENTIERS

CAHIER N° 145
mai 1997

A. SCHAAL [1]
H. M'SILTI [1]
P. TOLLA [1]

reçu : mars 1997.

[1] *LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16 (e-mail : {schaal,msilti,tolla}@lamsade.dauphine.fr).*

TABLE DES MATIÈRES

Pages

A hybrid method for general integer problems

Abstract

The hybrid method proposed in this paper combines interior point method, genetic algorithm and cut generation. The continuous optimization techniques quickly find points named "anchor points". The genetic algorithm explores the integer-solution space around the anchor points in order to find "satisfactory" solutions. Cuts enable one to determine "centered anchor points" inside the feasible space.

Keywords: general linear integer programming, interior point methods, genetic algorithm, economic cuts.

Une approche hybride de résolution de problèmes linéaires généraux en nombres entiers

Résumé

La méthode hybride proposée dans cet article combine une méthode intérieure «irréalisable», un algorithme génétique et l'exploitation de coupes économiques. La méthode intérieure trouve rapidement des solutions à composantes réelles appelées «points d'ancrage». L'algorithme génétique explore le voisinage de ces points d'ancrage afin de trouver des solutions réalisables à composantes entières «satisfaisantes». Les coupes permettent de trouver de nouveaux points d'ancrage «recentrés» situés à l'intérieur de l'espace admissible initial. Cette approche est présentée puis expérimentée sur 50 problèmes différents.

Mots-clés : programmation linéaire en nombres entiers, méthodes intérieures, algorithme génétique, coupes économiques.

1 Introduction

Cet article présente les performances de la méthode hybride [19] MIP-GENINT (Mathematical Integer Programming-GENetic-INTerior) destinée à résoudre des problèmes linéaires en nombres entiers généraux (IP_κ) :

$$(IP_\kappa)\begin{cases} \min & c^{\mathrm{T}}.x \\ s.c. & A.x + f = b \\ & 0 \leq x \leq u \\ & 0 \leq f \leq \kappa \\ & x \in \mathbb{Z}^n \\ & f \in \mathbb{R}^m \end{cases}$$

Les bornes (κ) portant sur les variables f permettent de prendre en compte des contraintes mal déterminées [19]. Les problèmes (LP) et (LP_{coupe}) ci-dessous, relaxations continues du problème (IP_κ), admettent des solutions dont les variables d'écart sont nulles. Ils permettent d'alléger la charge de travail de la méthode «entière» (phase 2.1 de MIP-GENINT) en vue de trouver des solutions «satisfaisantes».

$$(LP)\begin{cases} \min & c^{\mathrm{T}}.x \\ s.c. & A.x = b \\ & 0 \leq x \leq u \\ & x \in \mathbb{R}^n \end{cases} \qquad (LP_{coupe})\begin{cases} \min & c^{\mathrm{T}}.x \\ s.c. & A.x = b \\ & 0 \leq x \leq u \\ & c^{\mathrm{T}}.x = coupe \\ & x \in \mathbb{R}^n \end{cases}$$

Les difficultés rencontrées lors de la résolution du problème (IP_κ) sont les suivantes : - caractériser une solution réalisable à composantes entières,
- vérifier l'existence d'une solution réalisable à composantes entières,
- prouver l'optimalité d'une solution réalisable à composantes entières.

Le traitement de ces problèmes, que ce soit à l'aide de méthodes de coupes ou bien à l'aide de techniques de «Branch & Bound», nécessite un temps et une puissance de calcul incompatibles avec une utilisation dans le cadre d'un système interactif d'aide à la décision. Les méthodes de coupes mettent beaucoup de temps à trouver des solutions à composantes entières optimales du fait du nombre exponentiel de coupes à générer. Les coupes de Kelley, de Gomory ou de Fenchel sont des contraintes linéaires tenant compte des parties fractionnaires des solutions calculées par une méthode continue. Les méthodes énumératives souffrent de leur complexité théorique exponentielle et de la nécessité d'examiner tous les nœuds existant afin de converger vers une éventuelle solution. De plus, elles doivent disposer de procédures d'évaluation très efficaces afin de se déplacer dans l'arborescence et trouver rapidement des solutions réalisables à composantes entières.

La méthode qui nous sert de référence dans nos expérimentations, MIP-BBS (Mixed Integer Programming Branch and Bounded Simplex), a été présentée par Williams [23] et est implantée dans CPLEX. C'est une méthode de branch & bound dans laquelle les «bornes», portant sur les variables, sont resserrées en analysant les solutions calculées par une méthode continue. La méthode du Simplexe est utilisée afin de résoudre le problème linéaire associé à chaque nœud examiné par MIP-BBS. L'utilisation de méthodes intérieures, dans cette méthode, est difficile du fait des problèmes numériques introduits par les coupes générées à chaque nœud.

La méthode que nous proposons, MIP-GENINT, s'appuie sur les techniques de satisfaction de contraintes [20] afin de satisfaire le «plus» de contraintes possibles dans un laps de temps réduit. Cette méthode n'impose pas de manipuler des solutions réalisables à composantes entières car elles sont difficiles et coûteuses à construire dans le cas des problèmes généraux (cf. fig. 2). Notre méthode se décompose en deux phases (cf. fig. 1) :

- phase 1 : calcul du premier point d'ancrage x_c^* (méthode intérieure, LP),

- phase 2 : - 2.1 : recherche d'une solution à composantes entières (algorithme génétique, IP_κ),
  - 2.2 : calcul des points d'ancrage «recentrés» x_r^* (méthode intérieure, LP_{coupe}).

Cette décomposition [19] permet de concentrer la recherche «entière» sur une zone «prometteuse» de l'espace des solutions réalisables. Elle évite de se limiter à l'examen d'une seule zone dont on ne peut assurer qu'elle contient les points recherchés (cf. fig. 2 et 5).

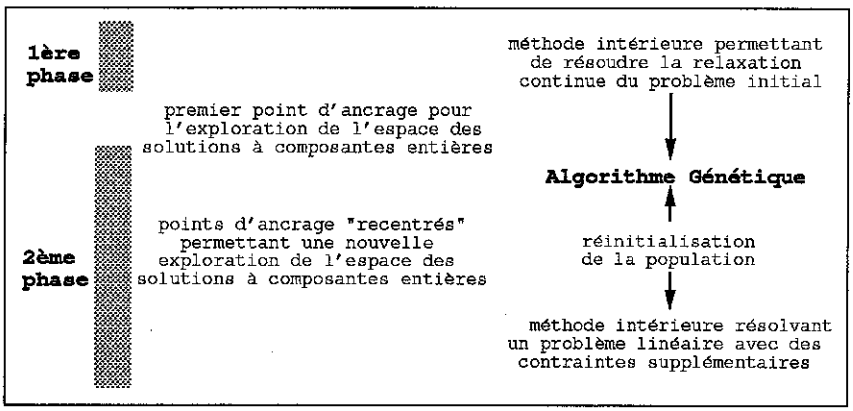

FIG. 1 – *méthode hybride intérieure et algorithme génétique*

L'étude que nous présentons est basée sur l'application d'un protocole expérimental validant l'approche proposée en la comparant à la méthode MIP-BBS. Les problèmes, générés aléatoirement, contiennent de 50 à 1 000 variables de 50 à 95 contraintes et leur matrice de contraintes est pleine (près de 70% des coefficients de la matrice sont différents de zéro). La méthode MIP-GENINT a été appliquée avec succès aux problèmes de 1 000 variables, en trouvant des solutions satisfaisant plus de 95 % des contraintes et en limitant, à moins de 4%, l'écart à la réalisabilité. Ce papier est organisé de la façon suivante: la section 2 présente la méthode hybride et les techniques utilisées (algorithme génétique, méthode intérieure, coupes économiques, recentrage). La section 3 présente les résultats établissant, expérimentalement, l'intérêt de la méthode (problèmes évalués, limites de MIP-BBS, intérêt et limites du premier point d'ancrage lors de l'initialisation de la population, rôle de l'opérateur de réinitialisation, impact des points d'ancrage recentrés). Les tableaux de résultats sont regroupés en annexe.

2 Approche hybride

La plupart des méthodes existantes mettent en œuvre un schéma hybride composé d'une phase «continue» et d'une phase «entière». Ces méthodes utilisent, en général, un grand nombre d'appels de la phase «continue» sans laisser beaucoup d'autonomie à la technique «entière». Celle-ci est utilisée uniquement afin de déduire, des solutions continues, les conditions que doivent éventuellement vérifier les solutions à composantes entières.

Une différence importante entre MIP-GENINT et les méthodes citées dans l'introduction réside dans la fréquence d'appel de la méthode «continue» (implantée dans la phase 2.2 de MIP-GENINT). Dans MIP-GENINT, ce n'est qu'après la conver-

gence de la méta-heuristique vers une solution non satisfaisante qu'une nouvelle coupe est générée puis qu'un nouveau point d'ancrage x_r^* est calculé. En effet, l'analyse de x_c^* ne permet pas de trouver, sauf dans le cas où la matrice définie par les contrainte est totalement unimodulaire, de solutions dans le cas général. Il existe des méthodes d'arrondi polynomiales pour des problèmes de couverture et de «packing»; dans le cas général, on ne connaît pas de méthodes qui le soient [11]. Les coupes sont générées en tenant uniquement compte de la fonction économique; elles sont intégrées au problème LP_{coupe} résolu à l'aide d'une méthode intérieure qui n'impose pas de connaître de solution de départ réalisable. La convergence de la méthode intérieure peut être anticipée (primale réalisabilité) car les points d'ancrage trouvés sont destinés à être arrondis. L'examen de l'«optimalité» de la solution trouvée par MIP-GENINT a lieu en évaluant un majorant de l'écart à l'optimum estimé par $c^{\mathrm{T}}.(x - x_c^*)$.

Les techniques utilisées dans MIP-GENINT sont choisies à cause de leur application réussie à certains problèmes en nombres entiers. Les méthodes intérieures adaptées par Portugal *et al.* [18] ont surpassé les méthodes spécialisées sur des problèmes de flots. Les algorithmes génétiques ont été utilisés par Michalewitcz *et al.* [14] afin de résoudre des problèmes de satisfaction de contraintes en variables réelles. Meguenni *et al.* [13] ont présenté une application des algorithmes génétiques à des problèmes d'optimisation en nombres entiers industriels de grande taille. Le lecteur peut se référer aux bibliographies publiées par Stewart *et al.* [21] et par Osman *et al.* [17] pour trouver des applications de cette méta-heuristique dans lesquelles la construction de solutions réalisables à composantes entières est triviale.

Les solutions sont améliorées de proche en proche (phase 2.1 de MIP-GENINT) afin de satisfaire les contraintes définies dans le modèle. Nous utiliserons des solutions vérifiant presque toutes les contraintes (plus de 95%) et suffisament «proche» des contraintes non satisfaites. Nous proposons une mesure de proximité relative appelée «taux de satisfaction». L'algorithme génétique utilisé permet d'explorer l'espace des points à composantes entières dans le voisinage (noté $V(\bullet)$) des points x_c^* et x_r^* (pavé grisé dans la fig. 2) et au-delà si nécessaire. Cet algorithme est adapté à l'importance de $\mid V(x) \mid = 2^n$, à l'irréalisabilité des points ainsi construits et à l'éventuel éloignement du point optimal à composantes entières [22].

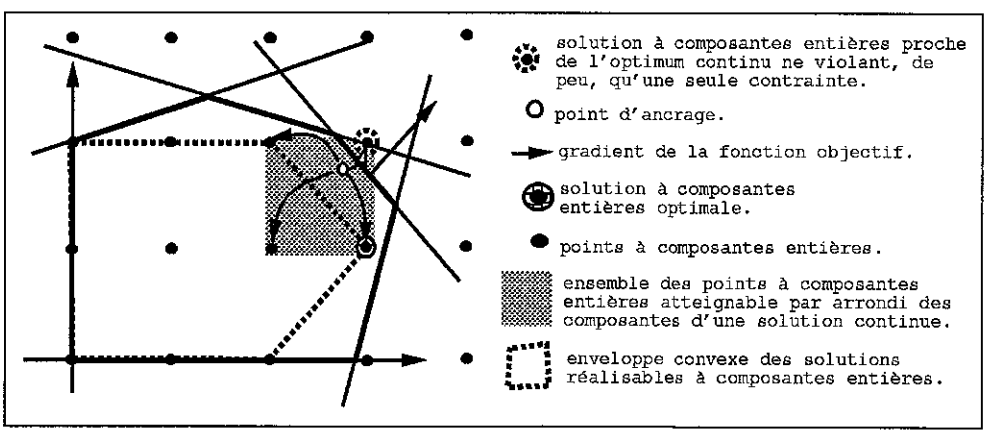

FIG. 2 – *reconstruction des sommets à composantes entières*

2.1 Algorithme génétique

Les algorithmes génétiques, dont les premières applications aux problèmes d'optimisation sont dues à Bethke [2], De Jong [6] et Holland [5], sont issus d'une

analogie avec l'évolution des espèces. Ces méthodes permettent d'alléger les hypothèses de régularité portant sur la fonction objectif et sur le domaine défini par les contraintes. Elles sont adaptées aux fonctions à optimiser ayant plusieurs extrema locaux et/ou mal définies et ont permis de résoudre de nombreux problèmes en nombres entiers. Ces algorithmes, de nature probabiliste, permettent de dégrader l'objectif ("hill-climbing") à la suite des déplacements locaux dans l'espace des solutions. L'algorithme génétique implanté dans MIP-GENINT est présenté dans la figure 3 ; il se distingue par l'opérateur de réinitialisation qui permet d'intensifier la recherche autour des points d'ancrage.

```
DÉBUT
    k ← 0
    choisir P^0                                          {Initialisation de la Population}
    RÉPÉTER
        P^{k+1} ← ∅                                      {Filtrage de la Population}
        POUR tout individu i de P^k FAIRE
            SI f(i) est suffisamment performant ALORS
                P^{k+1} ← P^{k+1} ∪ {i}                  {Altération de la Population}
        Croiser des individus appareillés aléatoirement
        Muter des individus choisis aléatoirement
        SI P^{k+1} spécialisée ou bien pas suffisament de progrès ALORS
            Réinitialiser P^k                            {Réinitialisation de la Population}
        k ← k + 1
    JUSQU'À ( test d'arrêt vérifié )
FIN
```

FIG. 3 – *variante de l'algorithme génétique incluant la réinitialisation*

Plusieurs individus forment une population P dont la taille reste fixe ; nous avons choisi un effectif faible de 10 individus dans MIP-GENINT. Chaque individu représente une solution du problème à l'aide d'un codage, basé sur un alphabet fini, qui permet de former son chromosome. Le chromosome est composé d'informations élémentaires, les gènes, qui ne peuvent prendre qu'un certain nombre de valeurs, les allèles. Le codage binaire présente l'intérêt de résumer efficacement l'information contenue en une position du génotype de l'individu. Cependant, à l'instar de Michalewicz *et al.* [15], nous utilisons directement les composantes entières du vecteur représentant l'individu. La capacité de chaque individu à résoudre le problème est évaluée par une fonction d'adaptation ; celle-ci est principalement basée sur le nombre de contraintes satisfaites dans MIP-GENINT. Un opérateur de reproduction de la population provoque le remplacement des individus les moins performants. Ce principe d'évolution est mis en œuvre par une «loterie» tenant compte de l'adaptation de l'individu.

Les individus i sont modifiés, à l'aide d'opérateurs de mutation et de croisement, par des déplacements dans leur voisinage $V(i)$. Ces transformations (cf. fig 4) ont lieu selon des lois de probabilité et peuvent tenir compte de l'adaptation de l'individu et de la nature du problème à résoudre si cela n'est pas trop coûteux. L'opérateur de croisement permet de propager, d'un individu à l'autre, des fragments de chromosome. Les couples de parents sont constitués en tenant compte de l'adaptation de chacun d'eux. L'opérateur de mutation permet de diversifier la population en altérant une partie du chromosome d'un individu. L'étude de paramétrage de MIP-GENINT a montré qu'une probabilité de mutation égale à 0,2 et une probabilité de croisement égale à 0,1 permettent de converger correctement [19]. En outre, elle a montré l'intérêt des opérateurs de mutation altérant, aléatoirement, un seul gène de l'individu courant en tenant compte du dernier point d'ancrage proposé.

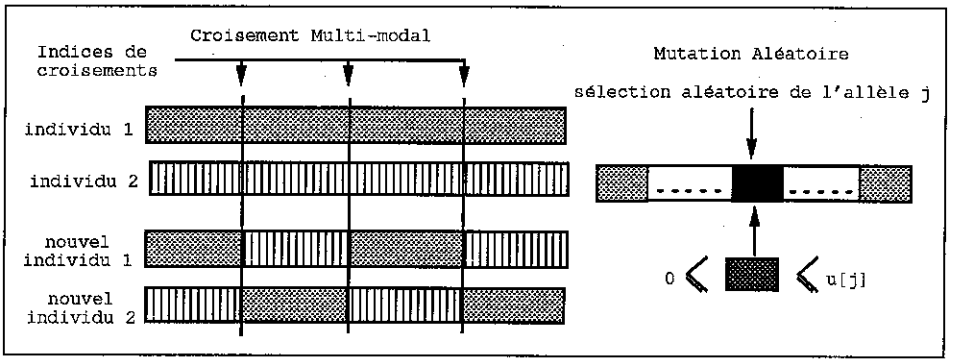

FIG. 4 – *opérateur de croisement et de mutation*

2.2 Méthode intérieure

Karmarkar [9], Ramakrishnan *et al.* [7], Karmarkar *et al.* [10] utilisent des méthodes intérieures et des techniques d'arrondi pour résoudre des problèmes totalement unimodulaires. L'heuristique d'arrondi, proposée par Karmarkar, tient compte de la paire optimale de solutions (x^*, y^*) du problème primal et du dual associé. Ceci permet de déterminer si les parties fractionnaires des composantes de x^* sont significativement différentes de 0 ou de 1. L'ouvrage édité par Beasley [1] présente d'autres aspects liés à l'utilisation des méthodes intérieures en programmation en nombres entiers et les travaux de Mitchell [16] présentent leur application dans les techniques de génération de coupes et ceux de Merle *et al.* [3] comparent une méthode intérieure à la méthode de Kelley.

Les méthodes intérieures imposent aux variables du problème d'être strictement positives en introduisant, dans la fonction objectif, une «barrière logarithmique» où un paramètre μ gère l'éloignement des frontières du polyèdre. Le problème initial est remplacé par une suite de problèmes modifiés résolus à μ fixé. La suite formée par les solutions optimales de ces problèmes converge, lorsque μ tend vers 0, vers la solution optimale du problème initial [8]. Les points générés décrivent une trajectoire située à l'intérieur du polyèdre convexe formé par $A.x = b$ (cas des méthodes réalisables cf. fig. 5) en imposant, le cas échéant, de se situer sur le chemin central défini[1] par $X.z = \mu.e$. Dans le cas «irréalisable», l'introduction de termes d'écartfig. dans le système de Karush-Kuhn et Tucker permet de converger dans le sous-espace défini par $A.x = b$ en un voisinage du point optimal.

La recherche d'une direction améliorante impose de résoudre un système de m équations à m inconnues. Cette étape implique des calculs en virgule flottante très coûteux et constitue la phase «critique» de ce type de méthode. Les versions «irréalisables» autorisent une moindre précision numérique dans cette phase, l'erreur commise étant prise en compte dans les vecteurs d'irréalisabilité recalculés à chaque itération. Les pas de déplacement, calculés à l'aide de recherches monodimensionnelles, sont multipliés par des coefficients strictement inférieurs à 1 (0,95 et 0,99995). Ceci garantit que les points calculés sont suffisamment éloignés des frontières du polyèdre et que la convergence de la méthode n'est pas ralentie par des déplacements trop faibles. Nous avons opté, dans MIP-GENINT, pour une méthode primale-duale «irréalisable» issue de la méthode de prédiction-correction de Mehrotra [12].

1. Si $x \in \mathbb{R}^n$, alors $X = diag(x) = \begin{pmatrix} x[1] & & 0 \\ & \ddots & \\ 0 & & x[n] \end{pmatrix} \in \mathcal{M}_{(n,n)}(\mathbb{R})$, $\mathrm{e} = (1, \cdots, 1)^{\mathrm{T}}$.

2.3 Recentrage et coupe économique

La coupe économique $c^{\mathrm{T}}.x \leq \alpha \times c^{\mathrm{T}}.x_c^*$, paramétrée par α, trouve plusieurs points d'ancrage selon le problème résolu (cf. fig. 5) :

- une solution extrême du polyèdre «coupé» (solution 1 ou 4),
- un centre «analytique» du polyèdre «coupé» (solution 2 ou 3).

Le paramètre α représente la «profondeur» de la coupe ; il est augmenté au fur et à mesure de la recherche avec un pas fixé à 1,005 dans MIP-GENINT. Dans le cas

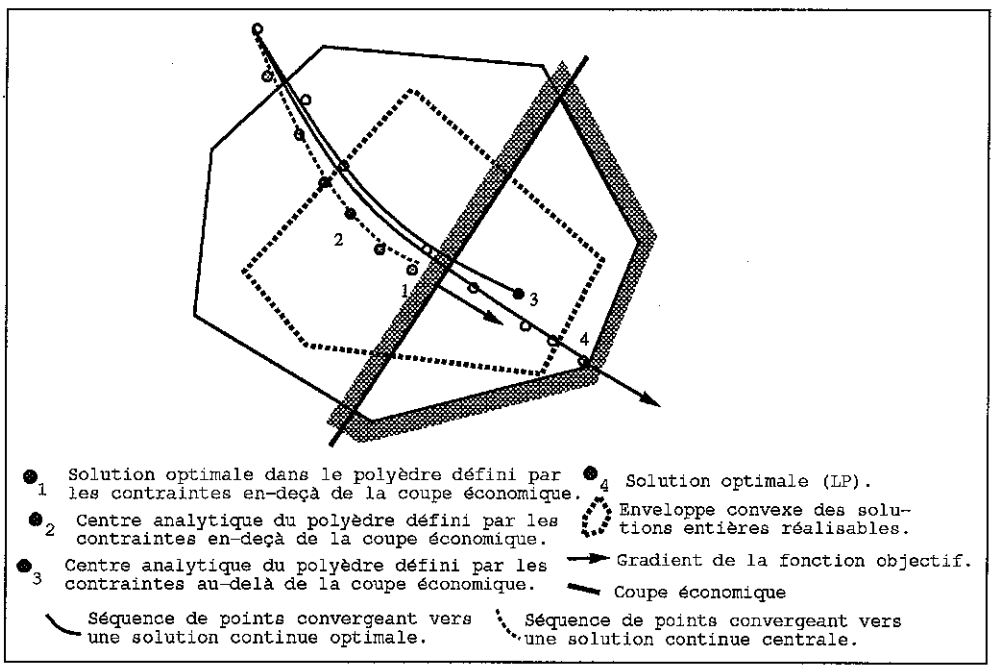

FIG. 5 – *Recentrage et coupe économique*

où l'on calcule un centre analytique, on peut chercher des solutions qui satisfont *au moins*, ou bien *au plus*, la limite imposée dans la coupe. Le calcul du centre analytique s'avère numériquement délicat et peu stable avec la méthode employée [19]. Par contre, la coupe économique induit une dégénérescence du problème LP_{coupe} se traduisant par la convergence de la méthode intérieure au centre de la face optimale. Pour cette raison, le calcul d'une solution «extrême» permet de trouver un point d'ancrage situé à l'intérieur du polyèdre initial.

L'introduction de la coupe économique peut produire un problème irréalisable difficile à gérer. Cette situation est détectée dans la méthode primale-duale quand l'irréalisabilité primale et/ou le facteur de recentrage «augmentent» inconsidérément. Dès qu'une coupe «irréalisable» est détectée, on diminue la «profondeur» de la coupe (α est divisé par 2 dans MIP-GENINT). L'examen de l'espace des solutions est alors recommencé à partir du premier point d'ancrage.

3 Résultats

Le tableau 1 présente les problèmes linéaires en nombres entiers généraux utilisés dans l'évaluation. Tous possèdent, par construction, une solution (x, f) au problème

(IP_0). Afin de tenir compte du caractère aléatoire de MIP-GENINT, nous avons créé dix instances, appelées dat0 à dat9, de chaque problème et nous avons évalué plusieurs fois chaque instance. Les critères d'évaluation, présentés dans le tableau 2, sont basés sur l'analyse des contraintes, des variables d'écarts f et du résidu r défini par : $r_j = \max\{(A.x - b)_j, 0\}, \forall j \in 1..m$. A contrario des problèmes de la collection Miplib maintenue a l'université de Rice, dont 72,88% des éléments sont à variables binaires et contiennent des systèmes de contraintes creux, nos problèmes sont très denses et ne possèdent que des variables entières à «grandes bornes».

3.1 Initialisation de la population

Cette expérience présente les performances de différents opérateurs d'initialisation de la population et n'utilise pas la réinitialisation. Certaines instances de problèmes possèdent un premier point d'ancrage très proche de solutions entières réalisables. P100.dat1 possède un premier point d'ancrage satisfaisant 58 contraintes sur 60 avec une norme 1 du résidu égale à 2. P100.dat0 possède un premier point d'ancrage ne satisfaisant que 54 contraintes sur 60 mais avec une norme 1 de résidu égale à 6. Cette observation encourage l'utilisation de ce point d'ancrage. L'opérateur, init0, initialise aléatoirement la population dans $[0, u]$; les autres opérateurs initialisent la population dans différents voisinages du premier point d'ancrage x_c^*. Les deux opérateurs init2 et init20 explorent le voisinage de x_c^* en tenant compte de l'importance de la partie fractionnaire de ses composantes. Chaque gène j (composante) de l'individu est fixé à $E(x_c^*)$ le cas échéant ou bien est tiré au sort parmi : $E(x_c^*[j]) + 1$, $E(x_c^*[j]) - 1$ et $E(x_c^*[j])$ (init2 ne peut tirer au sort $E(x_c^*[i])$). Les performances des opérateur init0, init2 et init 20 sont présentées dans le tableau 4. L'opérateur «init3» ne tenant pas compte de la partie fractionnaire de x_c^* et choisissant aléatoirement, pour chaque gène j de l'individu, une valeur dans : $E(x_c^*[j]) + 1$, $E(x_c^*[j])$ et $E(x_c^*[j]) - 1$ s'est avéré insuffisant.

3.2 Réinitialisation de la population

Les différents opérateurs de réinitialisation évalués utilisent :

- un redémarrage «à froid» aléatoirement dans $[0, u]$,
- un redémarrage «tiède» à partir du premier point d'ancrage issu de (LP),
- un redémarrage «à chaud» à partir de la meilleure solution à composantes entières, notée «x_e^*», trouvée jusque là.

L'opérateur R13 exploite «x_e^*» en choisissant aléatoirement, pour chaque gène j de l'individu, une valeur dans $x_e^*[j] - 1$, $x_e^*[j]$ et $x_e^*[j] + 1$. L'opérateur R33 agit de même en considérant le **premier** point d'ancrage x_c^* calculé tandis que l'opérateur R21 réinitialise aléatoirement dans $[0, u]$ la population. Les résultats obtenus, en termes de nombre de contraintes satisfaites, montrent un comportement assez stable de l'algorithme génétique quel que soit l'opérateur utilisé. Par contre, la mesure du résidu passe pour le problème le plus volatile, P100.dat8, de 200 à 1 200 unités. L'utilisation des opérateurs de réinitialisation permet d'augmenter les performances de MIP-GENINT de 1, 08 contraintes en moyenne ; le tableau 5 présente les performances obtenues avec MIP-GENINT sur P100 avec et sans réinitialisation.

3.3 Recentrage

Cette expérience, menée avec l'opérateur d'initialisation init2, l'opérateur de réinitialisation R33 et l'opérateur de mutation mut11, montre l'effet du recentrage sur les performances atteintes par MIP-GENINT en 10 000 générations. Le gain observé est, en moyenne, de 1,63 contraintes satisfaites ; la moyenne du taux d'insatisfaction, pour les instances de P100, tombe à 1,59 %. L'écart entre coût désiré et coût de la meilleure solution atteinte est de l'ordre de 10 %. L'intérêt de cette technique de recentrage réside dans les différentes mesures de l'écart constaté qui diminuent spectaculairement et dans la régularité de ces résultats. Les progrès obtenus se retrouvent dans l'augmentation, systématique, du nombre de contraintes satisfaites mais aussi dans le «taux de satisfaction» moyen qui atteint le millième quel que soit le problème résolu ; la figure 6 présente, avec une échelle logaritmique, les résultats moyens sur toutes les instances des différents problèmes. Les résultats sont présentés dans les tableaux 6 pour P50, 7 pour P100, 8 pour P200, 9 pour P400 et 10 pour P1000. La réinitialisation, ainsi que les points d'ancrage recentrés ont permis a MIP-GENINT de converger vers une solution du problème (IP_0) pour neuf instances de P50 avec un écart à l'optimalité présenté dans le tableau 11.

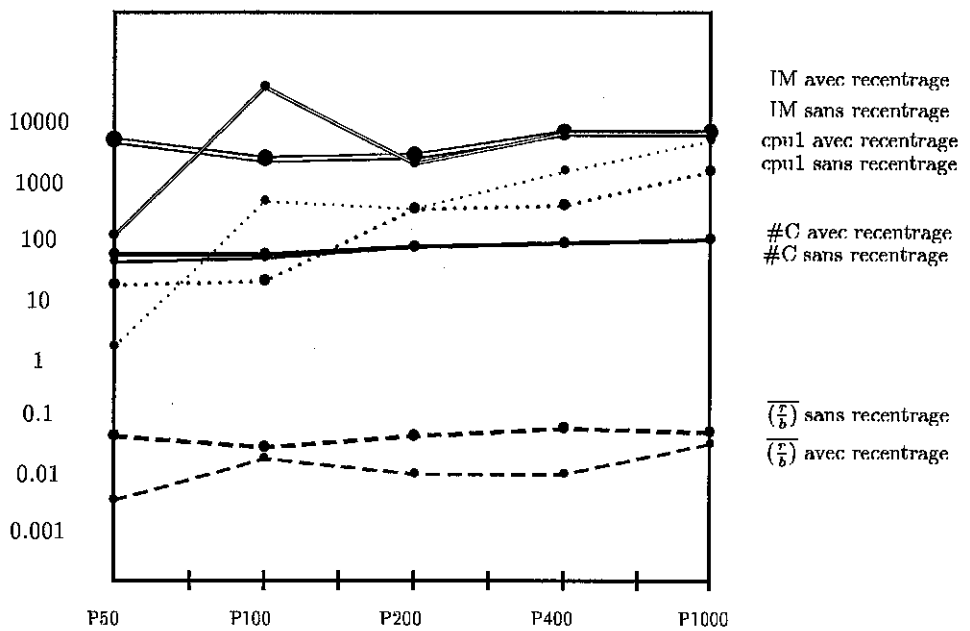

FIG. 6 – *intérêt du recentrage*

3.4 Limites de MIP-BBS

Afin d'évaluer l'efficacité de MIP-GENINT, le tableau 3 présente les performances de MIP-BBS dans le cas où les variables d'écart f sont soumises à des bornes finies pour P1000, P100 et P50. Lors de l'évaluation de MIP-BBS, CPLEX a consommé, avant de dépasser le nombre limite de nœuds (20 000 par défaut), près de 14.488 sec. c.p.u. et 816.776 itérations pour la première instance (dat0) du problème P400 (notée P400.dat0), près de 900 sec. c.p.u. pour les instances de la famille P100 et plus de 300 sec. c.p.u. pour chaque instance de P50. Hormis pour l'instance P50.dat4 (sur 30 problèmes évalués, MIP-BBS n'a pu fournir de solutions **à composantes**

entières. Les contraintes de bornes imposées aux variables d'écart f ont été difficiles à prendre en compte pour MIP-BBS. La non intégrité de ces solutions, est de 89 pour P400, 56 pour P100 et 43 pour P50. MIP-BBS a atteint une IInf de 15, pour P100.dat0, après plusieurs heures de calcul. Le nombre de composantes non entières est mesuré par l'indice IInf (Integer Infeasibility) avec une tolérance à l'intégrité fixée, par défaut dans CPLEX, à 10^{-05}. Ces, mauvais, résultats motivent l'approche proposée dans MIP-GENINT.

4 Conclusion

La méthode MIP-GENINT permet, à l'aide des points d'ancrage recentrés qui élargissent la recherche locale, :

1. de fournir «rapidement» des solutions réalisables acceptables,
2. d'améliorer ces estimations afin d'atteindre un écart relatif de l'ordre du millième en satisfaisant plus de 95% des contraintes ,
3. de donner plusieurs solutions approchées «comparables». La comparaison s'effectue en termes d'écart à la réalisabilité et en termes de coût économique.

Du fait du faible impact de l'opérateur de croisement (pour certains problèmes), il semble intéressant d'évaluer, dans la phase 2.1, un recuit simulé, éventuellement couplé avec l'algorithme génétique ; ce travail est en cours de réalisation. Les autres voies de recherche permettant d'améliorer les performances de MIP-GENINT sont de :

- privilégier les opérateurs d'évolution «augmentés par la connaissance» [4],
- paralléliser la méthode entière implantée en seconde phase de MIP-GENINT.

Références

[1] J.E. BEASLEY, éditeur. *Advances in Linear and integer programming.* Oxford science publications, Oxford, October 1996. Oxford lecture series in mathematics and its applications, 4.

[2] A.D. BETHKE. « *Genetic algorithm and function optimizer* ». PhD thesis, University of Michigan, 1970.

[3] O. du MERLE, J-L. GOFFIN, et J-Ph. VIAL. « On the comparative behavior of Kelley's cutting plane method and the analytic center cutting plane method ». Rapport Technique G-96-12, Groupe d'études et de recherche en analyse des décisions, Écoles des Hautes Études Commerciales, École Polytechnique, Université McGill, Université du Québec à Montréal, Montréal, Canada, 1996.

[4] D.E. GOLDBERG. *Algorithmes génétiques, exploration, optimisation et apprentissage automatique.* Addison-Wesley, 1994.

[5] J.H. HOLLAND. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Harbor, 1975.

[6] K.A. DE JONG. « *An analysis of the behavior of a class of genetic adaptive systems* ». PhD thesis, University of Michigan, 1975.

[7] A.P. KAMATH, N. KARMARKAR, K.G. RAMAKRISHNAN, et M.G.C. RESENDE. « Computational experience with an interior point algorithm on the satisfiability problem ». Rapport Technique, Mathematical Sciences Research Center, AT&T Bell Laboratories, Murray-Hill, NJ 07974, 1989.

[8] N. KARMARKAR. « A polynomial-time algorithm for linear programming ». *Combinatorica*, 4:373–395, 1984.

[9] N. KARMARKAR. « An interior-point approach to NP-complete problems ». Rapport Technique, AT&T Bell Laboratories, Murray-Hill, NJ 07974, 1989. Extended Abstract.

[10] N. KARMARKAR, M.G.C. RESENDE, et K.G. RAMAKRISHNAN. « An interior point algorithm to solve computationally difficult set covering problems ». *Mathematical Programming*, 52(1):597–618, 1991.

[11] S. LAKSHMINARAYANAN et R. CHANDRASEKARAN. « A rounding algorithm for integer programs ». *Discrete Applied Mathematics*, 50:267–282, 1994.

[12] I.J. LUSTIG, R.E. MARSTEN, et D.F. SHANNO. « Interior point methods for linear programming: computational state of the art ». *Operations Research Society of America, Journal on Computing*, 6(1):1–14, Winter 1994.

[13] K. MEGUENNI, H. M'SILTI, A. SCHAAL, et P. TOLLA. « Optimisation de problèmes industriels de grande taille : Cas de la diversité des câblages électriques ». communication, FRANCORO'95 : Journées Francophones de Recherche Opérationnelle, MONS, Belgique, Juin 1995.

[14] Z. MICHALEWICZ et C.Z. JANIKOW. « Handling constraints in genetic algorithms ». Department of Computer Science, University of North-Carolina, Charlotte, NC-28223, USA, 1995.

[15] Z. MICHALEWICZ, G.A. VIGNAUX, et M. HOBBS. « A nonstandard genetic algorithm for the nonlinear transportation problem ». *Operations Research Society of America, Journal on Computing*, 3(4):307–316, Autumn 1991.

[16] J.E. MITCHELL. « Computational experience with an interior point cutting plane algorithm ». Rapport Technique, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY, 12180, February 24, 1997.

[17] I.H. OSMAN et G. LAPORTE. « Metaheuristic: A bibliography ». *Annals of Operations Research*, 63:513–623, 1996.

[18] L.F. PORTUGAL, M.G.C. RESENDE, G. VEIGA, et J.J. JÚDICE. « A truncated primal-infeasible, dual-feasible network interior point method ». Rapport Technique, AT&T Bell Laboratories, Murray-Hill, NJ 07974, November 16, 1994.

[19] A. SCHAAL. « Une approche hybride de résolution de problèmes linéaires en nombres entiers : méthodes intérieures et meta heuristiques », 1997. Thèse en cours de rédaction.

[20] T. SCHIEX, H. FARGIER, et G.VERFAILLIE. « Problèmes de satisfaction de contraintes valués », 1996. à paraître dans la Revue d'Intelligence Artificielle.

[21] B.S. STEWART, C-F. LIAW, et C.C. WHITE III. « A bibliography of heuristic research through 1992 ». *IEEE Transactions on Systems, Man and Cybernetics*, 24(2):268–293, February 1994.

[22] M. WERMAN et D. MAGAGNOSC. « The relationship between integer and real solutions of constrained convex programming ». *Mathematical Programming*, 51:133–135, 1991.

[23] H.P. WILLIAMS. *Model building in mathematical programming.* Wiley, N-Y, 1985.

Table des matières

Table des figures

famille de problème	nombre de variables	nombre de contraintes	u_{max}	c_{max}
P1000	1 000	95	100	100
P400	400	90	100	100
P200	200	80	100	100
P100	100	60	100	100
P50	50	50	100	100

TAB. 1 – *description des problèmes*

abréviation	signification
rés.	résultat de la méthode
DV	la méthode diverge et ne trouve aucune solution
CV	la méthode converge vers une solution
u_{max}	valeur maximum des bornes portant sur les variables principales x
c_{max}	valeur maximum des composantes du vecteur de coût
inst.	instance évaluée de chaque problème
$\#C$	nombre de contraintes satisfaites
$\| r \|_1$	amplitude de la «non satisfaction» du système des contraintes
$\#A$	nombre de modifications de la meilleure solution
$\overline{(\frac{r}{b})} = \frac{1}{m} \times \sum_{j=1}^{m} \frac{r_j}{b_j}$	«taux de satisfaction» des contraintes
$\| f \|_\infty$	valeur maximum des variables d'écart
IM	itération à laquelle le meilleur individu a été trouvé
cpu1	temps de calcul pour trouver la meilleur solution entière approchée
cpu2	temps de calcul pour dépasser le nombre maximum de générations

TAB. 2 – *abréviations utilisées*

inst.	rés.	κ	$c(x_c^*)$.	rés.	κ	$c(x_c^*)$.	rés.	κ	$c(x_c^*)$.
	problème P1000			problème P100			problème P50		
dat0	DV	100 000	n.c.	DV	15 000	114 107	DV	20 000	61 914
dat1	DV	100 000	726 760	DV	10 000	86 626	DV	24 000	60 803
dat2	DV	100 000	799 440	DV	46 000	115 129	DV	20 000	69 842
dat3	DV	100 000	738 223	DV	53 000	109 960	DV	17 546	63 506
dat4	DV	100 000	653 674	DV	65 000	127 865	CV	20 000	57 725
dat5	DV	100 000	710 752	DV	50 000	116 202	DV	20 000	63 333
dat6	DV	100 000	719 061	DV	15 000	94 706	DV	20 000	75 978
dat7	DV	100 000	646 595	DV	40 000	102 163	DV	10 000	56 707
dat8	DV	100 000	658 817	DV	40 000	100 860	DV	20 000	61 709
dat9	DV	100 000	743 095	DV	10 000	126 541	DV	20 000	65 770

TAB. 3 – MIP-BBS*: résultats obtenus*

opé.	#A	IM	#C	$\| r \|_1$	$\overline{(\frac{r}{b})}$	$\| f \|_\infty$	$c(x_c^*)$
init0	1,4	3 066,7	55,40	325,50	0,052891	11 332	109 229
init2	1,3	3 085,4	55,30	327,70	0,047566	12 404	106 725
init20	1,4	3 795,5	55,50	319,30	0,056038	13 614	103 790

TAB. 4 – MIP-GENINT *: P100-Initialisation: moyenne des performances mesurées pour dat0*

nombre de contraintes saturées			
inst.	sans redémarrage	avec redémarrage	$c(x_c^*)$.
dat0	54,7	57	114 107
dat1	58	58	86 662
dat2	54,8	55,3	115 129
dat3	53,6	54,6	109 960
dat4	53,3	55	127 865
dat5	54,7	55	116 202
dat6	52,7	54,6	94 706
dat7	53,3	55,9	102 163
dat8	54,25	54,08	100 860
dat9	56	56,67	126 541

TAB. 5 – MIP-GENINT : *P100-Réinitialisation : effet du redémarrage*

inst.	#A	IM	#C	$\|\| r \|\|_1$	$\overline{(\frac{r}{b})}$	$\|\| f \|\|_\infty$	$c(x_e^*)$	$c(x_c^*)$.	cpu1	cpu2
dat0	2	628	49	1	0,000667	96	61 902	61 914	14,93	56,48
dat1	4	247	50	0	0	0	60 845	60 803,3	19,43	54,28
dat2	5	1549	50	0	0	0	69 952	69 841,7	13,79	54,59
dat3	5	500	50	0	0	0	63 727	63 506,1	12,16	53,14
dat4	3	128	50	0	0	0	57 852	57 724,9	13,81	56,36
dat5	8	74	50	0	0	0	63 511	63 332,6	12,35	58,67
dat6	7	529	50	0	0	0	76 386	75 977,6	4,93	50,52
dat7	6	492	50	0	0	0	56 807	56 705,6	38,53	59,34
dat8	0	0	50	0	0	0	61 886	61 813,4	0	65,26
dat9	5	6	50	0	0	0	65 984	65 770	10,04	51,67

TAB. 6 – MIP-GENINT : *P50-Recentrage demandé*

inst.	#A	IM	#C	$\|\| r \|\|_1$	$\overline{(\frac{r}{b})}$	$\|\| f \|\|_\infty$	$c(x_e^*)$	cpu1	cpu2
dat0	5	30 104	59	72	0,004364	4 667	112 240	451,06	1 439,89
dat1	1	13 405	59	1	0,000141	238	86 395	175,39	1 364,02
dat2	45	80 803	59	127	0,011199	13 303	118 100	1174,18	1 448,63
dat3	10	10 702	59	117	0,026712	10 375	103 343	151,24	1 438,52
dat4	10	4 716	58	133	0,026883	7 059	120 505	66,94	1 407,21
dat5	6	8 303	58	106	0,041549	4 578	112 714	121,74	1 384,48
dat6	10	47 402	59	41	0,007593	4 211	89 489	652,50	1 375,16
dat7	14	78 604	59	86	0,012045	4 674	102 697	1095,22	1 402,60
dat8	8	5 605	59	52	0,036111	6 046	95 283	87,53	1 449,49
dat9	3	36 807	59	2	0,000694	256	126 319	527,66	1 419,60

TAB. 7 – MIP-GENINT : *P100-Recentrage demandé*

inst.	#A	IM	#C	$\| r \|_1$	$\overline{(\frac{r}{b})}$	$\| f \|_\infty$	$c(x_e^*)$	cpu1	cpu2	$c(x_c^*)$.
dat0	11	5 902	79	254	0,005188	27 501	181 186	191,56	307,75	173 949
dat1	6	759	78	336	0,011240	22 663	210 582	33,91	292,09	209 466
dat2	6	174	78	194	0,012082	28 775	151 509	22,64	284,18	163 864
dat3	13	878	78	362	0,009582	29 758	197 819	39,29	317,18	197 384
dat4	13	2 501	78	236	0,013143	25 830	187 109	87,81	291,03	184 814
dat5	4	2 105	78	564	0,021147	62 523	148 578	82,76	296,24	166 255
dat6	8	4 009	78	33	0,003573	3 695	198 735	135,71	296,82	197 569
dat7	3	2	78	56	0,005099	4 332	200 531	20,98	294,64	201 161
dat8	8	2 289	78	242	0,013105	23 468	171 478	85,91	285,38	167 897
dat9	6	3 901	79	217	0,007351	31 671	200 806	134,27	305,40	206 297

TAB. 8 – MIP-GENINT : *P200-Recentrage demandé*

inst.	#A	IM	#C	$\| r \|_1$	$\overline{(\frac{r}{b})}$	$\| f \|_\infty$	$c(x_e^*)$	cpu1	cpu2	$c(x_c^*)$.
dat0	14	9 625	89	501	0,023194	58 067	383 168	967,60	981,84	348 043
dat1	4	5 402	88	533	0,008025	55 458	402 109	620,72	981,46	388 219
dat2	9	1 006	88	101	0,006310	8 879	324 130	188,67	915,60	331 991
dat3	27	8 707	89	681	0,015474	54 746	378 377	810,57	902,53	354 172
dat4	4	1 001	88	51	0,001404	4 359	367 667	197,16	972,51	366 981
dat5	20	9 506	89	341	0,009666	45 041	355 236	909,80	929,37	325 471
dat6	14	8 427	89	652	0,020698	349 662	167 196	11 441,50	12 889,43	n.c.
dat7	18	9 402	89	384	0,014175	57 723	374 195	914,78	937,41	348 132
dat8	18	9 802	88	311	0,006293	53 637	340 209	1 043,77	1 051,26	314 587
dat9	14	5 602	89	371	0,016229	42 065	400 372	564,07	907,03	361 448

TAB. 9 – MIP-GENINT : *P400-Recentrage demandé*

inst.	#A	IM	#C	$\| r \|_1$	$\overline{(\frac{r}{b})}$	$\| f \|_\infty$	$c(x_e^*)$	cpu1	cpu2	$c(x_c^*)$.
dat0	0	0	87	2828	0,084211	1.305 639	0	0	2 918,53	n.c.
dat1	6	1 212	92	396	0,009010	28 458	710 935	708,37	1 403,16	726 760
dat2	16	8 306	91	784	0,044094	82 405	922 429	7 886,70	8 992,18	799 440
dat3	15	9 901	92	1162	0,062877	71 845	886 311	8 112,58	8 125,62	738 223
dat4	7	8 404	93	947	0,022324	85 780	786 639	8 408,69	9 599,45	653 674
dat5	18	9 608	91	1130	0,071981	97 896	821 374	8 685,62	8 734,22	710 752
dat6	6	2 310	92	449	0,013588	33 610	705 644	798,78	1 397,02	719 061
dat7	18	9 495	92	1073	0,018339	66 527	752 408	2 406,83	2 503,87	646 595
dat8	2	9 702	93	71	0,001709	7 644	653 604	2 467,36	2 522,30	658 817
dat9	3	5 923	92	341	0,023868	42 698	726 294	1 089,28	1 409,57	743 095

TAB. 10 – MIP-GENINT : *P1000-Recentrage demandé*

inst.	majorant de l'écart à l'optimalité
dat1	0,00068582
dat2	0,00157929
dat3	0,00347841
dat4	0,00220182
dat5	0,00281687
dat6	0,00537527
dat7	0,00178818
dat8	0,00117450
dat9	0,00325376

TAB. 11 – MIP-GENINT : *P50: majorant de écart à l'optimalité pour* (IP_0)