RAIRO. RECHERCHE OPÉRATIONNELLE

P. SORIANO
M. GENDREAU

Fondements et applications des méthodes de recherche avec tabous

RAIRO. Recherche opérationnelle, tome 31, n° 2 (1997), p. 133-159
<http://www.numdam.org/item?id=RO_1997__31_2_133_0>

Recherche opérationnelle/Operations Research
(vol. 31, n° 2, 1997, pp. 133-159)

FONDEMENTS ET APPLICATIONS DES MÉTHODES DE RECHERCHE AVEC TABOUS (*)

par P. SORIANO ([1]) et M. GENDREAU ([1])

Communiqué par Dominique de WERRA

Résumé. – *Parmi les nouvelles heuristiques générales développées au cours des dernières années dans le domaine de l'optimisation combinatoire, la méthode de recherche avec tabous (RT) due à Glover (*tabu search *en anglais) s'est avérée très prometteuse ayant été appliquée avec succès à de nombreux problèmes difficiles. La RT est en fait une méta-heuristique qui combine une procédure de recherche locale avec un certain nombre de règles et de mécanismes permettant à celle-ci de surmonter l'obstacle des optima locaux tout en évitant de cycler. Dans le cadre de cet article, nous présentons une description des principes de la méthode et de ses diverses composantes. On y passe également en revue les principales applications de la RT réalisées à ce jour.*

Mots clés : Recherche avec tabous, algorithmes approximatifs, optimisation combinatoire.

Abstract. – *Among the new general heuristics developed over recent years in the field of combinatorial optimization, the tabu search method (TS) due to Glover has been found to be very promising having been succesfully applied to several difficult problems. TS is in fact a meta-heuristic which combines a local search procedure with a number of rules and mechanisms that enable it to overcome the obstacle of local optimality while preventing it from cycling. In this paper, we present a description of the principles underlying the TS method and of its various elements. A review of the main applications of TS to date is also provided.*

Keywords: Tabu search, approximate algorithms, combinatorial optimization.

1. INTRODUCTION

Depuis de nombreuses années, le domaine de l'optimisation combinatoire suscite un grand intérêt de la part des chercheurs opérationnels. En effet, celui-ci comprend quantité de problèmes qui, bien que très répandus dans

(*) Reçu en juin 1994.

([1]) Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale centre-ville, Montréal, Québec, Canada H3C 3J7.

les situations réelles, sont souvent extrêmement difficiles à résoudre de manière exacte. C'est notamment le cas du célèbre problème du voyageur de commerce (PVC). Lorsque l'on considère en particulier des instances de taille comparable à celles rencontrées dans la pratique, les méthodes de résolution traditionnelles telles les méthodes de coupes ou de séparation et évaluation progressive (*branch-and-bound* en anglais) s'avèrent rapidement prohibitives en termes de temps de calcul.

Afin de pouvoir malgré tout traiter ces problèmes, on doit donc faire appel à des approches de résolution approximatives : on troque alors la garantie d'optimalité présente dans les méthodes exactes contre l'espérance d'obtenir une solution de « bonne qualité » en un temps « raisonnable ». De nombreuses heuristiques ont ainsi été développées au fil des ans. Bien qu'elles soient généralement spécialisées à un problème en particulier, la plupart d'entre elles peuvent être réparties en deux familles distinctes. La première regroupe les approches de type constructif, comme par exemple les algorithmes « gloutons », qui *construisent* leur solution un élément à la fois, sans jamais remettre en question les choix passés. Ces méthodes sont en règle générale très rapides, mais la qualité des solutions qu'elles fournissent laisse souvent à désirer. La deuxième famille d'heuristiques classiques regroupe quant à elle les méthodes d'exploration locale (ou amélioration itéative), comme les méthodes d'échanges du type r-opt [60] ou Or-opt [65] pour le PVC, par exemple. Ce genre d'approche utilise comme point de départ une solution réalisable du problème et cherche à obtenir de meilleures solutions en effectuant une séquence de modifications locales qui *améliorent* chacune la solution en main, induisant ainsi une variation monotone de la fonction objectif. Ces approches-ci sont en règle générale plus puissantes que les heuristiques constructives, mais également beaucoup plus coûteuses en termes de ressources informatiques.

Toutefois, ces deux familles de méthodes partagent la même carence fondamentale : elles sont incapables de progresser au-delà du premier optimum local rencontré. Or les problèmes d'optimisation combinatoire comportent typiquement de nombreux optima locaux dont la valeur « économique » peut être fort éloignée de celle du ou des optima globaux. Ceci limite donc considérablement la puissance de ces approches. Pour faire face à cette situation, plusieurs heuristiques de type nouveau ont été développées ou adaptées à ce contexte au cours des dix dernières années, dont notamment le *recuit simulé*, les *algorithmes génétiques* et les méthodes de *recherche avec tabous*. C'est d'ailleurs la redécouverte en 1983 par Kirkpatrick *et al.* [56] de la méthode du recuit simulé et son application à l'archétype des problèmes d'optimisation combinatoire, le PVC, qui ont relancé cet intérêt pour de nouvelles stratégies de recherche. Cette méthode, originalement due à Metropolis *et al.* [62], s'inspire des phénomènes de recuit thermodynamique. Les algorithmes génétiques, initialement proposés en 1966 par Fogel *et al.* [23] puis étendus par Holland [52] en 1977, reposent quant à eux sur une analogie avec l'évolution des espèces. Enfin, les méthodes de recherche avec tabous, formalisées en 1986 par Glover [32] mais dont l'origine peut être retracée à 1977 [31], sont liées à des concepts issus du domaine de l'intelligence artificielle. Ces trois approches, bien qu'exploitant chacune des analogies et des arguments différents, permettent toutes de surmonter l'obstacle de l'optimalité locale et sont donc des alternatives particulièrement attrayantes par rapport aux algorithmes approximatifs traditionnels. Pour un survol de celles-ci *voir* [67].

Parmi ces heuristiques, la RT s'est révélée particulièrement efficace ayant été appliquée avec succès à de nombreux problèmes difficiles. Mais en quoi consistent les méthodes de RT et sur quoi se basent-elles ? Le but de cet article est de tenter de répondre à ces questions de manière simple et précise. Nous tenons toutefois à préciser que ceci ne se veut pas une description complète de la RT ou un mode d'emploi détaillé. Pour ce genre de traitement, nous renvoyons le lecteur aux articles fondamentaux de Glover [33, 35] ou aux excellents « modes d'emploi » que sont [40, 51 et 82]. Il n'est pas non plus question d'effectuer une analyse exhaustive de toutes les applications de la RT publiées dans la littérature scientifique. Il s'agit plutôt d'une sorte d'introduction visant à permettre au lecteur de se familiariser avec cette famille de méthodes et d'en apprécier la puissance et la flexibilité, tout en lui fournissant une liste assez complète des principales contributions méthodologiques et applications réalisées à ce jour.

Le plan de l'article s'établit comme suit. Dans la section 2, nous présenterons les concepts qui sont à la base des méthodes de RT ainsi que les principaux éléments qu'on y retrouve. Les notions essentielles seront illustrées à l'aide d'un exemple d'application de la RT au problème de k-coloration de graphes. Certains concepts plus « avancés », développés dans le but d'améliorer encore cette approche, seront décrits dans la section 3. Nous présenterons ensuite une revue synthétique des principales applications de la RT. Enfin, la section 5 qui conclut cet article, donnera également un aperçu des avenues de recherche futures dans ce domaine actuellement en plein essor.

2. FONDEMENTS ET ÉLÉMENTS DE BASE

2.1. Principes de la RT

Développée par Glover [32] et, indépendamment, par Hansen [41] sous l'appellation de *steepest ascent mildest descent*, la RT est en fait une métaheuristique. Elle consiste en effet en un ensemble de règles et de mécanismes généraux qui ont comme fonction de contrôler et de guider une heuristique interne, spécifiquement adaptée au problème à résoudre, afin de lui permettre de transcender l'obstacle des optima locaux. A la différence des algorithmes génétiques ou du recuit simulé qui font appel à des arguments probabilistes pour atteindre cet objectif, la RT exploite quant à elle la notion de *mémoire.* L'idée de base s'inspire de techniques de recherche utilisées en intelligence artificielle. Elle consiste à garder la trace du cheminement passé du processus de recherche dans une ou des mémoires et à se servir de cette information afin d'en orienter le déroulement futur.

Plus formellement et en adoptant une terminologie et une notation similaires à celles employées par Glover *et al.* [40] et Hertz *et al.* [48], on peut voir la RT comme une généralisation des méthodes d'amélioration locale traditionnelles. En effet, celle-ci explore itérativement l'espace X des solutions d'un problème d'optimisation donné, en se déplaçant d'une solution courante s, $s \in X$, à une nouvelle solution s' située dans le voisinage de s, $s' \in N(s)$, le choix de cette nouvelle solution s' étant déterminé par l'évaluation d'une certaine fonction objectif f. Le voisinage $N(s)$ d'une solution s est quant à lui défini comme l'ensemble des solutions $s' \in X$ pouvant être obtenues suite à l'application d'une transformation « locale » à s. Pour le PVC une telle transformation pourrait être un échange 2-opt, par exemple.

Toutefois, contrairement aux méthodes itératives classiques qui s'arrêtent dès qu'il n'y a plus de s' permettant d'améliorer f (c'est-à-dire, dans un contexte de minimisation, dès que $f(s') \geq f(s)$, $\forall s' \in N(s)$), ici s' est défini comme étant la « meilleure » solution parmi les éléments de $N(s)$ sans référence aucune à $f(s)$. Ceci permet à la RT de poursuivre la recherche de solutions meilleures même si cela entraîne une dégradation de la fonction objectif.

Cette modification du processus d'exploration rend donc la RT insensible aux optima locaux, mais elle introduit du même coup le risque de cycler dès que l'on sort de l'un de ces optima, en y retournant aussitôt. C'est là que la notion de mémoire trouve sa justification. Pour régler ce problème, la

RT conserve des informations sur le cheminement récent effectué à travers X afin d'interdire certaines transformations de la solution courante qui pourraient ramener la procédure vers des solutions déjà rencontrées. Ces transformations sont alors déclarées *tabou*, d'où le nom de la méthode. Ceci a pour effet de restreindre l'accès à certaines solutions dans $N(s)$. C'est ce mécanisme d'interdictions, ou de tabous, dont nous reparlerons en détail sous peu, qui permet d'orienter l'exploration de X au fur et à mesure qu'elle se déroule. Toutefois, pour conserver à l'approche suffisamment de flexibilité, le caractère tabou de ces transformations ne doit pas être maintenu en permanence. En limitant la durée de vie des tabous, on permet à la méthode de remettre en question ses choix passés (une fois les risques de cyclage disparus) et ainsi de mener une exploration beaucoup plus large du domaine des solutions.

Dans la description que nous venons de faire des principes directeurs de la RT, nous avons introduit sans pour autant les détailler plusieurs concepts importants qui ont un impact considérable sur la mise en œuvre de ces méthodes. Nous allons maintenant les expliciter un peu plus et les illustrer à l'aide d'un exemple d'application au problème de k-coloration d'un graphe dû à Hertz et de Werra [49]. Brièvement, ce problème consiste à trouver dans un graphe $G = (V, E)$ une partition s de l'ensemble V des sommets en k sous-ensembles, $s = (V_1, V_2, \ldots, V_k)$, telle qu'aucune arête de E n'ait ses deux nœuds terminaux dans le même sous-ensemble V_i. Dans ce contexte-ci on définit l'ensemble des solutions X comme étant l'ensemble des partitions de V en k sous-ensembles (sans pour autant exiger que ces partitions soient des k-colorations). Soit $E(V_i)$ le sous-ensemble des arêtes de E dont les sommets terminaux sont tous deux dans V_i, les transformations qui permettent de passer d'une solution s à une autre s' (et qui définissent la structure de voisinage exploitée par la procédure) consistent à choisir un sommet x incident à une arête appartenant à l'un ou l'autre des $E(V_i)$. On transfère alors ce x de V_i à V_j, $j \neq i$ (c'est-à-dire qu'on change sa couleur) et on obtient ainsi une nouvelle solution. La fonction objectif utilisée dans cette application correspond au nombre d'arêtes dont les extrémités sont dans le même V_i (ont la même couleur) :

$$f(s) = \sum_{i=1}^{k} |E(V_i)|.$$

Dès qu'on obtient une solution s telle que $f(s) = 0$ on a en main une k-coloration de G.

Comme on peut le constater la structure de ce problème est assez simple et en fait donc un bon candidat pour illustrer les divers concepts de la RT.

2.2. Les tabous

La définition et la gestion des tabous jouent bien entendu un rôle primordial dans la RT. Comme nous l'avons indiqué plus haut, ceux-ci correspondent à une sorte de mémoire à court terme qui indique à la procédure où elle a déjà été, lui permettant donc de diriger son exploration vers des régions du domaine non encore visitées.

La façon la plus simple de mettre en œuvre cet élément de la méthode consiste à conserver une liste T, des dernières $|T|$ solutions rencontrées et à interdire à la procédure d'y retourner. Ce genre de liste de tabous élimine instantanément toute possibilité que la procédure se retrouve dans un cycle de longueur $|T|$ ou moins. On la gère comme une liste « circulaire », en éliminant à chaque itération le tabou le plus ancien pour faire place au nouveau. Toutefois, ce genre de liste ne représente généralement pas la meilleure façon de faire car elle peut rapidement s'avérer très encombrante et coûteuse à gérer étant donné la quantité d'information requise pour décrire complètement une solution.

La méthode la plus fréquemment utilisée consiste à définir les tabous en fonction des « transformations » qui permettent de se déplacer d'une solution à une autre. On garde alors plutôt une liste des $|T|$ dernières transformations effectuées ou de certaines de leurs caractéristiques et on interdit à la procédure de les inverser. La plupart du temps on note directement dans la liste les transformations inverses qu'on veut proscrire. Dans le contexte de notre exemple de k-coloration, si la transformation effectuée consiste à transférer x de V_i à V_j, notons cela par le triplet (x, i, j), alors on pourrait interdire la transformation inverse en plaçant dans la liste le triplet (x, j, i). Malheureusement, si une telle liste de tabous empêche de retourner immédiatement à la solution qu'on vient de quitter, elle n'élimine pas vraiment les risques de cyclage. En effet, rien n'empêche la procédure d'effectuer la séquence de transformations suivantes : (x, i, j), puis (x, j, k) et enfin (x, k, i). On se retrouve alors exactement avec la même solution s où $x \in V_i$ qu'on avait avant la première des trois transformations.

Pour éviter ce genre de phénomène, on va plutôt conserver un ou plusieurs attributs caractérisant la transformation qu'on vient d'effectuer (ou la solution qu'on vient de quitter). Dans notre exemple ce sera le couple (x, i) représentant le sommet qui change de couleur et l'indice de son ancienne couleur. On interdit à la procédure de retransférer x dans V_i. Clairement cette restriction est beaucoup plus forte que la précédente et atteint bien son objectif d'empêcher le cyclage. Notons également qu'il est tout à fait envisageable de définir plusieurs listes de tabous, pour conserver différents types d'information, et de les utiliser simultanément. Parmi les premiers exemples d'algorithmes de RT comportant plus d'une liste tabou on peut citer les applications de Friden *et al.* [25] au problème du stable maximum et de Gendreau *et al.* [30] à celui de la clique maximale.

2.3. Critère d'aspiration

Les tabous sont un mécanisme dont le but premier est d'empêcher le cyclage de la méthode, mais ceux-ci peuvent parfois s'avérer trop forts et ainsi restreindre inutilement l'exploration du domaine des solutions. En particulier, lorsque les tabous sont définis en fonction des transformations, ils n'interdisent pas seulement de retourner à la solution précédente, mais à tout un ensemble de solutions dont plusieurs peuvent ne pas avoir été visitées encore. Ainsi dans notre exemple, le tabou imposé par la paire (x, i) empêche d'accéder à toute solution dans laquelle le sommet x se voit attribuer la couleur $i\,(x \in V_i)$. Il est tout à fait possible que certaines de ces solutions non encore visitées soient très attrayantes. Il faut donc qu'il y ait un mécanisme inverse à celui des tabous qui permette de révoquer le statut tabou d'une transformation si son application à la solution courante permet d'atteindre une solution jugée intéressante, sans pour autant introduire un risque de cyclage dans le processus.

C'est le concept de critère d'aspiration (ou fonction d'aspiration) qui remplit ce rôle. De nombreux critères d'aspiration ont été proposés dont plusieurs extrêmement sophistiqués. Nous ne détaillerons ici que les deux plus répandus qui sont également les plus simples. Pour plus de détails sur ces mécanismes nous renvoyons le lecteur à [40] et surtout à [39]. Notons que dans le cas d'une liste tabou de solutions, le concept de critère d'aspiration n'est pas applicable car toute annulation de tabou amène la procédure à cycler.

Le critère d'aspiration le plus simple consiste à révoquer le statut tabou associé à une transformation si cela permet d'obtenir une solution qui est supérieure à la meilleure solution rencontrée jusqu'à présent. C'est évidemment un critère très sévère qui ne devrait pas être vérifié souvent. Il n'ajoute donc pas beaucoup de flexibilité à la méthode, mais lui évite de commettre des « oublis » flagrants lorsque ce genre de situation se présente.

La deuxième méthode utilise une fonction $A(z)$ déterminant le niveau d'aspiration associé à chaque valeur z de la fonction objectif f. Étant donné une solution courante s avec $f(s) = z$, la valeur de $A(z)$ représente un seuil à atteindre si on veut s'assurer de ne pas cycler. Ainsi, quand on considère une transformation tabou menant de s à une solution voisine s' de valeur $f(s')$, si $f(s') \leq A(z)$ dans un contexte de minimisation (pour la maximisation on inverse le sens de l'inégalité), alors on est sûr que cette solution n'a jamais été visitée auparavant et on peut révoquer le statut tabou de cette transformation sans risque. Ce genre de fonction s'applique à des problèmes pour lesquels les valeurs de la fonction objectif sont entières. On initialise $A(z)$ en posant $A(z) = z - 1$ pour toutes les valeurs de z, puis chaque fois qu'on effectue une transition de s à s' telle que $f(s') \leq A(f(s))$ on pose $A(f(s)) = f(s') - 1$. C'est cette fonction d'aspiration qui est utilisée dans notre exemple d'application.

2.4. Évaluation du voisinage

Étant donné une solution s les transformations qu'il est possible d'y apporter par l'entremise de l'heuristique interne définissent un voisinage de s identifié par $N(s)$. Normalement, le choix d'une nouvelle solution s' s'effectue en examinant chacune des solutions contenues dans $N(s)$ et en choisissant la meilleure d'entre elles qui ne soit pas tabou, c'est-à-dire dans notre notation

$$s' \in \arg\min \{ f(\bar{s}) \mid \bar{s} \in N(s) \text{ et } \bar{s} \text{ non tabou} \}.$$

Dans certains cas, il est possible de maintenir sans trop d'effort une liste triée des voisins de s et donc d'effectuer cette évaluation de manière implicite sans avoir à les énumérer. Toutefois si tel n'est pas le cas, l'évaluation complète de $N(s)$ à chaque itération peut rapidement devenir exorbitante du point de vue des ressources informatiques, en particulier lorsque la structure de voisinage utilisée est « large » ou que l'évaluation de chaque solution est coûteuse. Dans bon nombre d'applications, on se contente plutôt de considérer un sous-ensemble $N' \subset N(s)$ qu'on génère aléatoirement et à choisir s' parmi les solutions contenues dans N'.

En plus de réduire la quantité de ressources nécessaires, cet échantillonnage de $N(s)$ constitue de par sa nature aléatoire un mécanisme anti-cyclage supplémentaire qui vient compléter les listes tabous. Il peut ainsi permettre d'utiliser des listes plus courtes que lorsque la totalité du voisinage est considérée.

2.5. Algorithme de RT

Considérons le problème d'optimisation consistant à minimiser la fonction $f(s)$ sur un domaine X et désignons par s^* la meilleure solution rencontrée jusqu'à présent et par f^* la valeur de celle-ci, $f^* = f(s^*)$. De plus, soit k un compteur d'itérations et étant donné que le sous-ensemble des solutions $s' \in N(s)$ qui sont sous l'effet d'un tabou dépend de la trajectoire passée de l'algorithme, donc de l'itération k, on peut alors désigner par $N(s, k)$ l'ensemble des solutions admissibles à partir de s à l'itération k (c'est-à-dire les solutions non tabous ou dont le statut tabou a été révoqué).

Cette notation étant établie, on peut maintenant schématiser l'algorithme générique de la RT comme suit :

1. <u>Initialiser</u>

- choisir une solution initiale $s_0 \in X$;
- poser $s := s^* := s_0$, $f^* := f(s_0)$ et $k := 0$;

2. <u>Tant que</u> *critère d'arrêt* non satisfait <u>faire</u>

- $k := k + 1$;
- choisir $s \in \arg \min_{s' \in N(s,k)} \{ f(s') \}$;
- <u>si</u> $f(s) < f^*$ <u>alors</u> $s^* := s$ *et* $f^* := f(s)$;
- mettre à jour les tabous.

Différents critères d'arrêt peuvent être utilisés. Parmi ceux-ci les plus courants consistent à arrêter la procédure :

– dès qu'on a obtenu une solution optimale (lorsqu'on en connaît la valeur) ou une solution pour laquelle la fonction objectif atteint une valeur pré-déterminée ;

– après un certain nombre d'itérations ou un certain temps de calcul ;

– après qu'un certain nombre d'itérations aient été effectuées sans qu'il y ait eu d'amélioration de la meilleure solution trouvée s^*.

L'algorithme de RT que nous venons de décrire constitue en fait la version de base de cette approche, ainsi que le mentionne Glover dans [33] où elle est désignée sous l'appellation de *simple tabu search* en anglais. Bien que déjà capable de produire des résultats impressionnants, comme en font foi ses applications recensées à la section 4, là ne s'arrêtent pas les ressources de l'approche. En effet, d'autres concepts et de nombreux raffinements de ceux exposés jusqu'ici ont été proposés et expérimentés dans le but d'augmenter encore la puissance et la robustesse de ces méthodes. Dans la section qui suit nous aborderons ces concepts plus « avancés ».

3. COMPOSANTES SUPPLÉMENTAIRES ET RAFFINEMENTS

3.1. Intensification et diversification

Une des notions fondamentales sur laquelle repose la RT est, comme cela a été mentionné à la section précédente, la mémoire. Toutefois, dans la description que nous avons fait de la méthode jusqu'ici, son usage est relativement peu poussé. En effet, celui-ci se limite à un contrôle à court terme du déroulement de l'exploration, l'algorithme se contentant de « mémoriser », grâce aux listes de tabous, ce qu'il vient de faire *récemment* afin d'éviter de retourner vers des solutions ou des classes de solutions qu'il a déjà visitées.

Dans une version plus complète de l'approche, le rôle de la mémoire va au-delà de cette fonction première et influence également le processus de recherche à moyen et long terme. Ceci s'effectue par l'entremise de deux nouveaux concepts, soit l'*intensification* et la *diversification* de la recherche, qui n'exploitent plus des informations basées sur la proximité dans le temps (comme c'est le cas pour les mécanismes de tabous), mais plutôt d'autres types d'informations faisant appel à des notions de fréquence.

Plus précisément, le but de l'intensification consiste, comme son nom l'indique, à intensifier l'effort d'exploration dans certaines régions du domaine qui ont été identifiées comme étant prometteuses. On va donc analyser sur une plus longue période le cheminement de la procédure et essayer de déterminer si les « bonnes » solutions rencontrées jusqu'à présent n'auraient pas des caractéristiques communes, c'est-à-dire si certaines caractéristiques n'y apparaissent pas fréquemment. Dans l'affirmative, on procédera alors à une modification du processus d'exploration afin de favoriser leur présence dans les nouvelles solutions visitées par l'algorithme. Généralement, une telle modification est appliquée périodiquement et pour une durée limitée, après quoi le processus normal reprend son cours.

Plusieurs stratégies d'intensification peuvent être utilisées dépendant des contextes. La plus simple correspond à retourner à l'une des meilleures solutions rencontrées jusque-là, puis à reprendre l'exploration à partir de ce point en réduisant la longueur des listes tabous pour un nombre limité d'itérations [48]. Une autre façon consiste à fixer temporairement certaines portions de la solution courante associées par le passé à de nombreuses solutions de qualité supérieure (par exemple certains arcs du cycle dans le PVC), permettant ainsi à la procédure de se concentrer sur la portion du domaine définie par ces solutions partielles. Le même genre d'effet peut être obtenu en modifiant plutôt la fonction f servant à évaluer les solutions candidates afin de favoriser celles qui contiennent ces portions de solution au détriment des autres (par le biais d'un terme de pénalité par exemple). Enfin, il est aussi possible de remplacer pour un temps limité l'heuristique interne normalement utilisée par une autre méthode plus puissante ou encore d'intensifier la recherche en élargissant le voisinage réellement évalué à chaque itération (en augmentant par exemple la taille de l'échantillon considéré $N'(s)$). D'autres stratégies plus complexes exploitant plus à fond l'information générée par le processus d'exploration et divers concepts d'*apprentissage* sont décrites dans [37, 48, 58].

De son côté, le concept de diversification est en quelque sorte l'inverse de celui d'intensification. L'objectif visé est ici de rediriger la procédure vers des portions de l'espace des solutions où elle n'est pas encore allée (ou peu souvent) afin d'éviter que le processus de recherche ne soit trop localisé et laisse de grandes régions du domaine totalement inexplorées. La stratégie de diversification la plus simpliste est bien entendu d'interrompre périodiquement la procédure et de la faire recommencer à partir d'une nouvelle solution générée aléatoirement ou, un peu mieux, en choisissant celle-ci plus intelligemment de façon à être sûr de se retrouver dans une région non encore visitée.

Une autre manière de procéder consiste à biaiser la fonction d'évaluation en introduisant un terme qui pénalise les transformations effectuées fréquemment ou certaines caractéristiques jugées trop souvent présentes dans les solutions rencontrées jusque-là. Inversement on peut utiliser le même genre de technique dans le but de favoriser des caractéristiques rarement rencontrées. Il est intéressant de faire remarquer que ce type de stratégie de diversification peut d'ailleurs être utilisé de façon continue, tout au long de l'exploration, sans qu'il y ait besoin d'arrêter le processus. L'application de Gendreau *et al.* [29] est une des premières ayant fait appel à une telle *diversification continue.*

Enfin, ainsi que le présentent Hertz *et al.* [48], on peut voir les contraintes que doivent satisfaire les solutions d'un problème pour être réalisables comme autant de « montagnes » infranchissables. En relaxant certaines d'entre elles et en pénalisant leur violation, on « aplanit » en quelque sorte le relief de l'espace des solutions en réduisant la « hauteur » de ces montagnes. Il devient alors possible de franchir ces obstacles et de se déplacer beaucoup plus facilement vers d'autres régions. Les solutions visitées durant cette phase de diversification peuvent évidemment être irréalisables. Pour se ramener dans le domaine d'origine, on augmente graduellement les pénalités jusqu'à obtenir des solutions réalisables. Ce genre de technique a été employée avec succès dans le cadre d'applications à des problèmes d'horaires qui sont typiquement des problèmes comportant de nombreuses contraintes [10, 50]. Un concept similaire, l'*oscillation stratégique* (*strategic oscillation* en anglais), remonte aux travaux à l'origine du développement de la RT [31]. Initialement proposé dans un contexte assez différent de celui-ci, il peut toutefois être vu comme une généralisation de ces techniques de relaxation puisqu'il permet d'agir sur d'autres caractéristiques des solutions que leur admissibilité. Son utilisation dans le cadre d'applications à d'autres problèmes d'horaires [63] et à des problèmes de partitionnement de graphes s'est révélée fort utile [40].

Comme on peut le voir, l'intensification et la diversification sont des concepts complémentaires qui viennent élargir la gamme des outils disponibles pour contrôler et guider « intelligemment » la recherche de bonnes solutions. L'incorporation de ces éléments dans un algorithme de RT lui confèrent une flexibilité et une adaptabilité accrues qui se traduit généralement par une augmentation très significative de sa puissance et de sa robustesse. L'étude de Soriano et Gendreau [74] sur l'impact de différentes stratégies de diversification dans le contexte du problème de clique maximale présente une des premières illustrations des gains pouvant découler de l'addition de ce type de techniques.

3.2. Quelques raffinements

Bien d'autres améliorations, la plupart sous la forme de raffinements de certains aspects des éléments de base décrits à la section 2, ont été et continuent d'être apportées régulièrement par les nombreux chercheurs s'intéressant à ces méthodes. Bien que leur description détaillée dépasse le cadre de cet exposé nous allons brièvement en aborder quelques-unes qui nous paraissent parmi les plus utiles.

Gestion des listes de tabous

Tout d'abord, concernant la gestion des listes de tabous, nous avons vu que celles-ci étaient traditionnellement gérées comme des listes circulaires dont la taille était fixe et prédéterminée. Généralement cette taille est dépendante de celle de l'instance traitée et croît avec elle [33]. Toutefois, il n'existe pas de règle universelle permettant de déterminer une valeur appropriée de ce paramètre, il faut donc se baser sur l'expérimentation. La seule observation qui ait été faite à ce sujet est que la taille d'une liste ne doit être ni trop petite, car alors la méthode se met à cycler, ni trop grande, car dans ce cas trop de restrictions sont imposées à la procédure et celle-ci aura tendance à stagner. En effet, dans la RT ce sont les transformations les plus attrayantes qui sont exécutées en priorité, si la liste est trop longue celles-ci seront maintenues tabou plus longtemps forçant ainsi la procédure à considérer plus de solutions « médiocres » ce qui, dans un contexte de minimisation se traduit par un accroissement de la valeur moyenne des solutions visitées [40, 48]. Il est en général relativement aisé de déterminer un ordre de grandeur approprié et même un intervalle de « bonnes » valeurs, mais le choix d'une valeur précise donnant les meilleurs résultats est plus délicat.

Pour contourner ce problème d'ajustement, mais surtout pour impartir une plus grande robustesse des algorithmes par rapport aux instances particulières des problèmes à résoudre, plusieurs stratégies ont été proposées afin d'effectuer une gestion *dynamique* plutôt que *statique* des listes de tabous. Ainsi dans [73] et [38] leur taille est modifiée à intervalles réguliers et varie respectivement sur 4 et 3 valeurs distinctes, selon une séquence prédéterminée. La méthode utilisée dans [80] change elle aussi la taille périodiquement, mais la valeur de celle-ci est tirée aléatoirement dans $[T_{\min}, T_{\max}]$, où les deux bornes $T_{\min}$ et $T_{\max}$ sont des paramètres fournis à la procédure. Une généralisation de cette idée de taille aléatoire a été proposée et appliquée dans [29], où la durée du tabou associée à *chaque transformation* est tirée aléatoirement dans un intervalle similaire. Dans cette approche, la structure de liste disparaît pour être remplacée par des *étiquettes tabou* (*tabu tags* en anglais), directement associées aux transformations.

Enfin, d'autres stratégies de gestion exploitant des concepts différents de ceux qui précèdent ont également été proposées, notamment celles dans [42, 87], utilisant des tables de hachage (*hashing functions* en anglais), et dans [16, 35], basées sur des notions de logique séquentielle.

Évaluation du voisinage

Tournons-nous maintenant vers l'élément central de l'heuristique interne, soit le choix de la prochaine solution. Tel que mentionné plut tôt (2.4), l'évaluation complète du voisinage de la solution courante $N(s)$ s'avère souvent extrêmement coûteuse et ne constitue donc pas toujours une stratégie d'évaluation raisonnable. Dans ces situations, on a recours à des techniques d'échantillonnage aléatoire pour générer un sous-ensemble $N'(s)$ à partir de $N(s)$ et on restreint le choix de la prochaine solution à celles appartenant à $N'(s)$. On peut ainsi réduire considérablement les coûts d'évaluation.

Cette manière simple de gérer l'évaluation de $N(s)$ est assez répandue, mais elle n'est pas toujours des mieux adaptées au problème à résoudre.

En effet, si $N(s)$ ne contient que peu de « bonnes » solutions et beaucoup de « mauvaises », comme c'est le cas dans les problèmes de tournées de véhicules entre autres, un $N'(s)$ choisi aléatoirement n'a que peu de chances de contenir des solutions candidates intéressantes. Dans un tel contexte, cette stratégie risque plus de nuire que d'améliorer les performances de la méthode.

D'autres stratégies de gestion ont été proposées afin de faire reposer le choix de $N'(s)$ sur autre chose que le pur hasard [34, 39, 40]. Celles-ci, référées dans la littérature sous l'appellation de « candidate list strategies » soit *listes de candidats*, suscitent de plus en plus d'intérêt. Parmi les plus répandues on peut citer celles qui exploitent une sorte de décomposition cyclique du voisinage en sous-ensembles déterminés stratégiquement. A chaque itération, on ne considère qu'une portion de ceux-ci tout en s'assurant que les autres le seront au cours des itérations à venir. Les stratégies d'évaluation du voisinage utilisées dans [22, 57 et 71] sont de ce type. Une autre approche consiste à orienter la composition de $N'(s)$ en fonction de caractéristiques jugées attrayantes dans les solutions recherchées comme c'est le cas dans [29] et [66]. D'autres principes pouvant être exploités pour gérer l'évaluation de $N(s)$ sont également décrits dans [40] et surtout dans [34].

3.3. Mise en œuvre efficace

Ainsi qu'on aura pu le constater à la lecture des sections précédentes, la RT est une approche reposant sur des concepts qui sont très simples à la base. Toutefois, étant donné la quantité de composantes qu'elle peut comporter et la flexibilité dont on dispose pour les adapter au problème à résoudre, développer un algorithme de RT vraiment efficace n'est pas *a priori* une trivialité. En particulier, la nécessité de « calibrer » les nombreux paramètres employés par un tel algorithme peut sembler extrêmement délicat.

Bien qu'important, cet ajustement des mécanismes de l'algorithme n'est pas l'aspect primordial d'une mise en œuvre efficace. En effet, comme le soulignent très justement Hertz *et al.* [48], l'efficacité d'un algorithme de RT, et en général de toute approche de résolution, dépend principalement de la modélisation du problème, c'est-à-dire du choix de l'espace des solutions à explorer X, du voisinage $N(s)$ et de la fonction objectif f permettant de choisir la prochaine solution dans $N(s)$. Bref, elle concerne le choix de l'heuristique interne exploitée par l'algorithme de RT. Il y a souvent plusieurs possibilités pour effectuer ces choix et ceux-ci sont critiques car même la calibration la plus soigneuse des paramètres de contrôle ne peut espérer compenser un mauvais choix à ce niveau. A l'opposé, une « bonne » modélisation du problème rendra l'approche beaucoup moins sensible à la précision de ces ajustements.

Sans entrer trop dans les détails, mentionnons tout de même que la structure de voisinage choisie doit permettre de se déplacer facilement d'une solution à une autre et qu'il est souhaitable qu'elle garantisse qu'à partir d'une solution donnée s, il existe toujours un chemin menant de s à une solution optimale du problème [48]. A cette fin, il est tout à fait envisageable que le choix de l'espace des solutions exploré par la procédure ne soit pas le même que celui du problème à résoudre. En effet, il peut s'avérer presque impossible dans certains cas de définir une structure de voisinage en utilisant seulement les solutions du vrai problème. Dans ce genre de situation, il est souvent utile de relaxer certaines contraintes du problème original afin d'obtenir un voisinage plus praticable. L'exemple de k-coloration qui nous a servi auparavant, illustre bien cette idée puisque la contrainte forçant les arêtes à avoir leurs extrémités dans deux « couleurs » différentes est relaxée. Le voisinage des « partitions » qu'on obtient alors se trouve à être beaucoup moins difficile à explorer que celui des k-colorations.

Une autre remarque importante concerne le choix de la fonction objectif. Celle-ci devrait fournir assez d'information pour permettre de départager facilement les solutions candidates à chaque itération. Si la fonction objectif du problème initial ne varie que sur un petit intervalle et assigne les mêmes valeurs à de nombreuses solutions, présentant en quelque sorte de larges « plateaux », elle ne remplit évidemment pas ce critère et rend la recherche d'une solution optimale beaucoup plus ardue. On a dans ce cas intérêt à définir une fonction objectif *auxiliaire* qui conserve le pouvoir discriminant de celle du vrai problème, mais permet également de guider l'exploration. La fonction définie dans l'application au problème de k-coloration en est un bon exemple puisqu'elle permet d'identifier une solution optimale ($f(s) = 0$) tout en orientant la recherche vers les solutions intermédiaires les plus prometteuses, soit celles contenant le moins d'arêtes monochromatiques (ces solutions sont les moins irréalisables par rapport au vrai problème).

Comme on peut le constater, ces décisions sont primordiales et doivent se baser sur une analyse sérieuse du problème à résoudre. Pour une discussion plus poussée de ces notions d'efficacité de la modélisation en particulier et de la mise en œuvre des méthodes de RT en général, nous référons le lecteur intéressé à [40] et surtout à [48]. D'autres discussions intéressantes sur ces questions sont également présentées dans [33, 35, 37, 39, 51, 82].

TABLEAU I

Quelques applications de la recherche avec tabous

Problème traité	Référence	Voisinage		Listes	Intensification	Diversification
		Transformations	Evaluation	Nombre, Gestion		
Problèmes de graphes						
k-coloration	Hertz et de Werra [49] (1987)	insertions	échantillon	1, fixe	—	—
T-coloration	Costa [11] (1993)	insertions	échantillon	1, fixe	—	—
stable maximum/clique maximale	Friden *et al.* [25] (1989)	échanges	liste de candidats	3, fixe	—	—
	Gendreau *et al.* [30] (1993)	insertions	complète/ échantillon	2, fixe	—	—
	Soriano et Gendreau [74] (1993)	insertions	complète/ échantillon	2, fixe	—	pénalité sur transformations fréquentes/ redémarrage
Problèmes de tournées de véhicules						
PTV classique	Pureza et França [69] (1991)	insertions et échanges	complète	1, fixe	—	—
	Osman [66] (1993)	insertions et échanges	complète/ candidats	1, dynamique	—	—
	Gendreau *et al.* [29] (1994)	insertions GENI	liste de candidats	1, dynamique (étiquettes aléatoires)	élargissement du voisinage évalué	pénalité sur transformations fréquentes
	Taillard [81] (1993)	insertions	liste de candidats	1, dynamique	—	pénalité sur transformations fréquentes
PTV réels	Semet et Taillard [71] (1993)	insertions	liste de candidats (approximation)	1, dynamique	évaluation exacte	—
	Rochat et Semet [70] (1993)	insertions	complète	1, dynamique	fixation de composantes de la solution	—
PTV+fenêtres de temps	Potvin *et al.* [68] (1993)	échanges	liste de candidats	1, fixe	—	—

TABLEAU I (*suite*)
Quelques applications de la recherche avec tabous

Problème traité	Référence	Voisinage		Listes	Intensification	Diversification
		Transformations	Évaluation	Nombre, Gestion		
Problèmes d'ordonnancement						
Chaîne de traitement	Widmer et Hertz [83] (1989)	échanges	complète	1, fixe	—	—
	Taillard [79] (1990)	insertions	complète	1, dynamique		
Traitement à séquences fixées	Widmer [84] (1991)	insertions	complète	1, fixe	—	—
	Taillard [78] (1989)	inversion d'opérations	complète (approximation)	1, dynamique	—	pénalité sur transformations fréquentes
	Dell'Amico et Trubian [18] (1993)	échanges	complète	1, dynamique	—	—
Séquencement sur 1 machine	Laguna *et al.* [57] (1991)	échanges et insertions	liste de candidats	2, fixe	—	—
	Woodruff et Spearman [86] (1992)	insertions	variable	1, dynamique (hachage)	—	oscillation stratégique
Autres problèmes						
Affectation quadratique	Chakrapani et Skorin-Kapov [8] (1993)	échanges	complète	1, dynamique	fixation de composantes de la solution	pénalité sur variables fréquentes
Localisation/allocation multi-produits avec équilibrage	Crainic *et al.* [12] (1993)	échanges	échantillon (approximation)	2, fixe	—	redémarrage (fréquences)
Affectation d'appels dans un réseau avec capacités	Anderson *et al.* [1] (1993)	échanges	liste de candidats	1, fixe	évaluation complète du voisinage	—

4. APPLICATIONS

Les méthodes de recherche avec tabous constituent une technique de résolution encore bien jeune et la plupart des applications qui en ont été faites datent de moins de cinq ans. Leur nombre ne cesse toutefois de croître ce qui atteste sans équivoque de l'intérêt suscité par ses grandes possibilités. Dans cette section, nous allons effectuer un survol des principaux domaines où la RT a été appliquée. Nous tenons à rappeler cependant que cette liste ne se prétend aucunement exhaustive.

D'autre part, afin d'illustrer la variété tant des problèmes spécifiques traités que des mises en œuvre de l'approche, nous avons dressé dans le tableau I une liste de quelques applications choisies, en donnant à chaque fois une description schématique des principales composantes de l'algorithme de RT utilisé.

Une des premières applications de la RT a eu lieu dans le domaine des *problèmes de graphes*. C'est l'application au problème de k-coloration due à Hertz et de Werra [49] qui nous a servi plus tôt pour illustrer les concepts de la méthode et qui a été publiée en 1987. Cette application, bien que n'utilisant que les concepts de base de la RT, s'est révélée à l'époque extrêmement efficace étant capable de résoudre des problèmes comportant jusqu'à 1000 sommets et reste aujourd'hui encore l'une des meilleures. L'approche a été comparée à une application de recuit simulé que les auteurs avaient réalisée précédemment [9] et la RT s'est révélée nettement supérieure du point de vue qualité des solutions tout en nécessitant beaucoup moins de temps de calcul. Dans [20], une telle application sert à résoudre des sous-problèmes dans un algorithme de séparation et évaluation progressive pour le problème de *coloration minimale* d'un graphe. La méthode ainsi obtenue permet de résoudre exactement des problèmes allant jusqu'à 70 sommets pour des densités de 0,5.

Une généralisation du problème de coloration classique est traitée par Costa [11] soit celui de *T-coloration*. Ce problème qui consiste à trouver une coloration des sommets telle que les « couleurs » de deux sommets adjacents soit « distantes » d'au moins une certaine valeur (dépendante des deux sommets concernés) a des applications dans la gestion des bandes de fréquences de télécommunication en particulier. Dans cet article, des applications de RT et de recuit simulé sont développées et comparées. Ici encore, la RT s'avère performante et supérieure au recuit.

Toujours dans le domaine des graphes, deux adaptations différentes de la version de base de la RT aux problèmes équivalents de stable maximum, due à Friden *et al.* [25], et de clique maximale, par Gendreau *et al.* [30], se sont montrées très efficaces, avec un très léger avantage à la seconde. Les trois variantes de l'approche présentées dans cette dernière, ont aussi déclassé une procédure de recuit considérée jusque-là très performante. Ces heuristiques se sont ensuite vues nettement améliorer par l'addition de mécanismes de diversification et ces versions plus « modernes » de RT se sont montrées extrêmement performantes et robustes sur une large gamme de problèmes de toutes sortes [74, 75]. Bien que publiée en 1993 seulement, les heuristiques de [30] ont été développées en 1990 et sont avec celle de [25] parmi les premières applications de RT faisant appel à plusieurs listes de tabous. Cette façon de faire est depuis devenue monnaie courante. Enfin, l'heuristique pour le stable maximum citée un peu plus tôt a été intégrée dans une procédure exacte de séparation et évaluation progressive des plus performantes [26].

Un autre domaine fertile pour le développement d'algorithmes de RT est celui des problèmes de tournées de véhicules (PTV). Dans [66], Osman développe plusieurs variantes de RT et de recuit simulé pour la version du PTV avec contraintes de capacité sur les véhicules et de durée sur les tournées. La RT s'avère ici encore supérieure au recuit et de très bons résultats sont obtenus sur une série de 14 problèmes classiques de la littérature, améliorant à l'époque les meilleures solutions connues dans 7 cas.

Une autre application, due à Gendreau *et al.* [29], basée sur une définition de voisinage différente et exploitant cette fois les concepts de diversification et d'intensification, s'est avérée encore plus performante que la précédente, dominant toutes les méthodes classiques et obtenant ou améliorant la meilleure solution connue dans 11 des 14 problèmes de référence. Cette application est également intéressante car c'est la première où ont été proposés et utilisés les concepts d'étiquettes de tabous et de diversification continue décrits à la section 3.

On se doit de noter cependant qu'avant ces deux adaptations particulièrement réussies, deux autres tentatives s'étaient soldées par des résultats peu enthousiasmants [69, 85]. Ces premières tentatives sur le PTV se contentaient d'une application très « directe » des principes de base de la méthode, sans effort particulier de modélisation. De leur côté, les adaptations de Osman et Gendreau *et al.* sont beaucoup plus « travaillées ». Les résultats qu'elles obtiennent viennent donc souligner les remarques effectuées à la section précédente à l'effet qu'une mise en œuvre vraiment efficace de la méthode nécessitait une bonne connaissance du problème et un effort certain de modélisation.

Depuis, d'autres applications au PTV ont vu le jour. Tout d'abord Taillard [81], avec une implantation parallèle de RT basée sur un voisinage semblable à celui d'Osman, a réussi à obtenir la meilleure solution connue pour chacun des 14 problèmes de référence. D'autre part, deux applications récentes, dues à Semet et Taillard [71] et à Rochat et Semet [70], ont permis de traiter avec succès des instances réelles de PTV rendus encore plus complexes par l'addition de nombreuses contraintes supplémentaires. Dans les deux cas, les solutions fournies représentaient des économies de l'ordre de 15 % par rapport aux solutions traditionnelles. Ces deux applications illustrent bien la capacité de la RT à s'adapter au contexte des problèmes pratiques où de nombreuses contraintes viennent souvent s'ajouter au problème de base et en détruire la structure. Avec la RT, ces contraintes complicantes sont facilement prises en compte, généralement sous la forme de tabous (les solutions irréalisables sont déclarées tabous) ou de pénalités, permettant ainsi de restaurer la structure du problème de base qui peut alors être exploitée.

D'autres applications de RT à des problèmes connexes ont été menées à bien récemment, notamment pour le PTV avec fenêtres de temps [27, 28, 68] et pour le m-PVC avec objectif Minmax [24]. Enfin, plusieurs auteurs ont développé des algorithmes de RT pour le PVC classique avec un certain succès [22, 36, 61], surtout pour des instances de grande taille. Ces adaptations n'arrivent toutefois pas à battre les meilleures heuristiques de type classique sur les problèmes plus conventionnels.

Un troisième champ d'applications ayant vu de nombreuses adaptations de la RT est celui des problèmes d'ordonnancement, en particulier dans un contexte de production. Une des premières est celle de Widmer et Hertz [83] au problème de la chaîne de traitement dans un atelier de production, connu en anglais sous le nom de « permutation flow shop ». Dans ce problème n objets doivent subir des opérations sur une chaîne de traitement comportant m machines et on cherche à déterminer l'ordre dans lequel ce traitement devrait être effectué afin de minimiser le temps total écoulé entre le début du traitement du premier objet et la fin de celui du dernier, en anglais le *makespan*. Cette application qui n'utilise que les éléments de base de la RT s'est révélée supérieure aux meilleures heuristiques connues alors sur un ensemble de 50 problèmes. Une adaptation subséquente de Taillard [79] améliorant les résultats de celle-ci est venue confirmer l'avantage de la RT sur la plupart des heuristiques classiques utilisées pour ces problèmes. Enfin, Daniels et Mazzola [17] développent une procédure de RT basée sur une stratégie sophistiquée de recherche imbriquée pour traiter une généralisation du problème de chaîne de traitement. Sur une batterie de 1 600 problèmes leur approche surclasse les autres heuristiques testées. De plus, sur les 480 instances pour lesquelles la solution optimale était connue, la RT la trouve dans 70 % des cas tout en enregistrant un écart moyen de 0,3 % et une déviation maximale de 2,5 % seulement.

Dans [18], Dell'Amico et Trubian appliquent la RT au célèbre problème de traitement à séquences fixées ou *job shop scheduling* en anglais. Dans ce problème d'ordonnancement les n objets peuvent nécessiter différentes séquences d'opérations sur les m machines (éventuellement sans avoir besoin de chacune d'entre elles), mais ces séquences sont fixes et l'ordre de traitement des objets à chaque machine peut varier. On cherche un ordonnancement sur les m machines qui minimise l'intervalle de temps écoulé entre le début des opérations et la fin de la dernière opération sur le dernier objet. L'algorithme de RT développé surclasse les meilleures mises en œuvre de recuit simulé tant du point de vue qualité des solutions que temps de calcul, tout en se révélant extrêmement robuste ce qui n'est pas le cas de ses concurrentes. Une autre adaptation de la RT à ce problème, due à Taillard [78] s'était déjà révélée supérieure au recuit et a donné lieu à une version « parallèle » constituant probablement la meilleure heuristique actuellement disponible.

Enfin, de nombreuses autres adaptations de la RT qui méritent d'être signalées ont été publiées dans la littérature scientifique. Bien qu'il ne puisse être question de les décrire toutes ici, nous allons quand même en énumérer quelques-unes en les regroupant par problème ou secteur d'application pratique. On peut ainsi citer celles concernant : d'autres problèmes d'ordonnancement [2, 6, 7, 57, 58, 59, 63, 76, 84, 86], le problème d'affectation quadratique [8, 72, 73, 80], les problèmes de logique probabiliste incluant les problèmes de satisfaisabilité [42, 44, 53, 54], les problèmes de localisation [3, 12, 45], des problèmes en télécommunications [1, 38, 64], les réseaux de neurones [4, 82], les problèmes d'horaires [10, 46, 47, 50], et divers autres problèmes [5, 16, 19, 43, 55, 77].

5. CONCLUSION

La méthode de la recherche avec tabous dont nous avons décrit les principes dans cet article est, comme nous venons de le voir, une approche de résolution approximative qui s'est avérée très efficace sur une large gamme de problèmes d'optimisation combinatoire reconnus difficiles. Dans la plupart des cas, celle-ci surclasse facilement les méthodes heuristiques traditionnelles. De plus, lorsque comparée aux autres méthodes d'exploration locale modernes, et notamment au recuit simulé, celle-ci s'est montrée en règle générale nettement supérieure en produisant des solutions de meilleure qualité tout en requérant passablement moins de temps de calculs.

On doit également souligner l'extrême flexibilité et la grande adaptabilité de l'approche qui lui permet d'incorporer aisément les contraintes complicantes qu'on retrouve si souvent dans les problèmes réels et face auxquelles les méthodes traditionnelles sont sans recours. Celle-ci peut aussi être combinée de différentes façons avec d'autres méthodes d'optimisation plus classiques, entre autres sous la forme de sous-routine pour résoudre les sous-problèmes d'algorithmes de séparation et évaluation progressive (permettant ainsi d'obtenir des bornes de bonne qualité rapidement) ou encore pour définir des procédures hybrides comportant une composante RT et une composante heuristique classique.

Toutefois, comme il faut s'y attendre et ainsi que le démontrent certaines applications assez décevantes, la RT n'est pas une panacée permettant de tout résoudre facilement. De plus, certains de ses succès, comme ceux obtenus pour le problème de tournées de véhicules classique, ont été réalisés après plusieurs tentatives d'adaptation infructueuses et un travail de modélisation considérable. Nous tenons d'ailleurs à rappeler ici encore qu'il est essentiel d'avoir une bonne connaissance et compréhension du problème à résoudre afin de pouvoir développer une heuristique vraiment efficace.

Malgré les succès retentissants de l'approche dans la pratique et l'engouement sans cesse croissant qu'elle suscite, une ombre demeure sur ce tableau car on ne connaît que très peu de choses sur la RT d'un point de vue théorique. En effet, aucune démonstration mathématique prouvant la convergence de la méthode n'est encore connue à ce jour. Le seul résultat disponible est celui dû à Faigle et Kern [21] où ils démontrent la convergence asymptotique d'une version de recuit simulé modifiée pour ressembler à la RT. Cette variante de la RT est toutefois très éloignée des algorithmes effectivement réalisés. De plus, le type de théorème de convergence obtenu est somme toute de peu d'utilité puisqu'il ne garantit la convergence sur une solution optimale avec probabilité tendant vers 1 qu'après avoir écoulé un temps infini. Cette remarque que nous faisons sur la faiblesse de ces aspects théoriques n'a pas pour but de dénigrer l'efficacité pratique de la méthode, mais bien de rappeler qu'il y a encore beaucoup de travail à effectuer sur ce plan.

D'autres avenues de recherche très prometteuses s'ouvrent également aux chercheurs œuvrant dans ce domaine. Mentionnons tout d'abord l'application à ces méthodes des techniques informatiques de parallélisation. La RT est en effet une approche qui se prête bien à ce contexte informatique. Déjà plusieurs mises en œuvre parallèles ont été réalisées [8, 28, 61, 78, 80, 81] produisant d'excellents résultats et des études sur les diverses façons de paralléliser ces méthodes commencent à apparaître [13, 14, 15]. Enfin, le développement d'approches hybrides intégrant des principes tirés des diverses méthodes modernes d'exploration locale que sont le recuit simulé, les algorithmes génétiques et la recherche avec tabous semble des plus intéressants et pourrait éventuellement mener à des approches de résolution encore plus puissantes dans un proche avenir.

REMERCIEMENTS

Nous tenons à remercier le Fonds FCAR du Gouvernement du Québec qui a partiellement financé cette recherche.

BIBLIOGRAPHIE

1. C. A. Anderson, K. Fraugnaugh, M. Parker et J. Ryan, Path Assignment for Call Routing: An Application of Tabu Search, *Annals of Operations Research*, 1993, *41*, p. 301-312.
2. J. W. Barnes et M. Laguna, Solving the Multiple-Machine Weighted Flow Time Problem Using Tabu Search, *IIE Transactions*, 1992 (à paraître).
3. S. Benati et G. Laporte, Tabu Search Algorithms for the (r/x_p)-Medianoid and the (r/p)-Centroid Problems, *Location Science*, 1994, *2*, p. 193-204.
4. D. Beyer et R. Ogier, Tabu Learning: A Neural Network Search Method for Solving Nonconvex Optimization Problems, *Proceedings of the International Joint Conference on Neural/Network*, IEEE and INNS, Singapour, 1991.
5. J. A. Bland et G. P. Dawson, Tabu Search and Design Optimization, *Computer-Aided Design*, *23*, 1991, p. 195-202.
6. J. Bovet, C. Constantin et D. de Werra, A Convoy Scheduling Problem, à paraître dans *Discrete Applied Mathematics*, 1991.
7. P. Brandimarte, Routing and Scheduling in a Flexible Job Shop by Tabu Search, *Annals of Operations Research*, 1993, *41*, p. 157-184.
8. J. Chakrapani et J. Skorin-Kapov, Massively Parallel Tabu Search for the Quadratic Assignment Problem, *Annals of Operations Research*, 1993, *41*, p. 327-342.
9. M. Chams, A. Hertz et D. de Werra, Some Experiments with Simulated Annealing for Coloring Graphs, *European Journal of Operations Research*, 1987, *32*, p. 260-266.

10. D. Costa, A Tabu Search Algorithm for Computing an Operational Time Table, *European Journal of Operations Research*, 1994, *76*, p. 98-110.

11. D. Costa, On the Use of Some Known Methods for T-Coloring of Graphs, *Annals of Operations Research*, 1993, *41*, p. 343-358.

12. T. G. Crainic, M. Gendreau, P. Soriano et M. Toulouse, A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements, *Annals of Operations Research*, 1993, *41*, p. 359-384.

13. T. G. Crainic, M. Toulouse et M. Gendreau, Towards a Taxonomy of Parallel Tabu Search Algorithms, *INFORMS J. on Computing* (à paraître).

14. T. G. Crainic, M. Toulouse et M. Gendreau, *A Study of Synchronous Parallelization Strategies for Tabu Search*, publication CRT-934, Centre de recherche sur les transports, Université de Montréal, 1993.

15. T. G. Crainic, M. Toulouse et M. Gendreau, An Appraisal of Asynchronous Parallelization Approaches for Tabu Search Algorithms, *Annals of Operations Research*, 1996, *63*, p. 277-299.

16. F. Dammeyer et S. Voβ, Dynamic Tabu List Management Using the Reverse Elimination Method, *Annals of Operations Research*, 1993, *41*, p. 31-46.

17. R. L. Daniels et J. B. Mazzola, A Tabu Search Heuristic for the Flexible-Resource Flow Shop Scheduling Problem, *Annals of Operations Research*, 1993, *41*, p. 207-230.

18. M. Dell'Amico et T. Trubian, Applying Tabu Search to the Job-Shop Scheduling Problem, *Annals of Operations Research*, 1993, *41*, p. 231-252.

19. W. Domschke, P. Frost, et S. Voβ, Tabu Search Techniques for the Quadratic Semi-Assignment Problem, *New Directions for Operations Research in Manufacturing*, G. Fandel, T. Gulledge et A. Jones (Eds.), Springer, 1991, p. 389-405.

20. N. Dubois et D. de Werra, EPCOT: An Efficient Procedure for Coloring Optimally with Tabu Search, *Computers and Mathematics with Applications*, 1993, *25*, p. 35-45.

21. U. Faigle et W. Kern, Some Convergence Results for Probabilistic Tabu Search, *ORSA Journal on Computing*, 1992, *4*, p. 32-37.

22. C. N. Fiechter, A Parallel Tabu Search Algorithm for Large Scale Traveling Salesman Problems, *Discrete Applied Mathematics*, 1994, *51*, p. 243-267.

23. L. J. Fogel, A. J. Owens et M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, Wiley, New York, 1966.

24. P. M. França, M. Gendreau, G. Laporte et F. M. Müller, The *m*-Traveling Salesman Problem with Minmax Objective, *Transportation Science*, 1995, *29*, p. 267-275.

25. C. Friden, A. Hertz et D. de Werra, STABULUS: A Technique for Finding Stable Sets in Large Graphs with Tabu Search, *Computing*, 1989, *42*, p. 35-44.

26. C. Friden, A. Hertz et D. de Werra, TABARIS: An Exact Algorithm Based on Tabu Search for Finding a Maximum Independent Set in a Graph, *Computers and Operations Research*, 1990, *17*, p. 437-445.

27. B. L. Garcia, *Développement de techniques de recherche tabou pour le problème de tournées de véhicules avec fenêtres de temps*, publication CRT-931, Centre de recherche sur les transports, Université de Montréal, 1993.

28. B. L. Garcia, J.-Y. Potvin et J.-M. Rousseau, A Parallel Implementation of the Tabu Search for the Vehicle Routing Problem with Time Windows, *Computers & Operations Research*, 1994, *21*, p. 1025-1033.

29. M. Gendreau, A. Hertz et G. Laporte, A Tabu Search Algorithm for the Vehicle Routing Problem, *Management Science*, 1994, *40*, p. 1276-1290.

30. M. Gendreau, P. Soriano et L. Salvail, Solving the Maximum Clique Problem Using a Tabu Search Approach, *Annals of Operations Research*, 1993, *41*, p. 385-404.

31. F. GLOVER, Heuristics for Integer Programming Using Surrogate Constraints, *Decision Science*, 1977, *8*, p. 156-166.

32. F. GLOVER, Future Paths for Integer Programming and Links to Artificial Intelligence, *Computers and Operations Research*, 1986, *5*, p. 533-549.

33. F. GLOVER, Tabu Search – Part I, *ORSA Journal on Computing*, 1989, *1*, p. 190-206.

34. F. GLOVER, *Candidate List Strategies and Tabu Search*, CAAI Research Report, University of Colorado, Boulder, 1989.

35. F. GLOVER, Tabu Search – Part II, *ORSA Journal on Computing*, 1990, *2*, p. 4-32.

36. F. GLOVER, Multilevel Tabu Search and Embedded Search Neighborhoods for the Traveling Salesman Problem, *ORSA Journal on Computing*, 1991 (à paraître).

37. F. GLOVER, Tabu Search for Nonlinear and Parametric Optimization (with Links to Genetic Algorithms), *Discrete Applied Mathematics*, 1991 (à paraître).

38. F. GLOVER et M. LAGUNA, Bandwidth Packing: A Tabu Search Approach, *Management Science*, 1993, *39*, p. 17-29.

39. F. GLOVER et M. LAGUNA, Tabu Search, dans *Modern Heuristic Techniques for Combinatorial Problems*, C. R. REEVES (Ed.), Blackwell, Oxford, 1993, p. 70-150.

40. F. GLOVER, E. TAILLARD et D. DE WERRA, A User's Guide to Tabu Search, *Annals of Operations Research*, 1992, *41*, p. 3-28.

41. P. HANSEN, *The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming*, présenté au *Congress on Numerical Methods in Combinatorial Programming*, Capri, Italie, 1986.

42. P. HANSEN et B. JAUMARD, Algorithms for the Maximum Satisfiability Problem, *Computing*, 1990, *44*, p. 279-303.

43. P. HANSEN, B. JAUMARD et E. DA SILVA, Average Linkage Divisive Hierarchical Clustering, *Journal of Classification*, 1992 (à paraître).

44. P. HANSEN, B. JAUMARD et M. POGGI DE ARAGAO, *Mixed Integer Column Generation Algorithms and the Probabilistic Maximum Satisfiability Problem*, Proceedings of the 2nd Integer Programming and Combinatorial Optimization Conference, Carnegie-Mellon, 1992.

45. P. HANSEN, E. DE LUNA PEDROSA FILHO et C. CARNEIRO RIBEIRO, Location and Sizing of Offshore Platforms for Oil Exploration, *European Journal of Operations Research*, 1992, *58*, p. 202-214.

46. A. HERTZ, Tabu Search for Large Scale Timetabling Problems, *European Journal of Operations Research*, 1991, *54*, p. 39-47.

47. A. HERTZ, Finding a Feasible Course Schedule Using Tabu Search, *Discrete Applied Mathematics*, 1992, *35*, p. 255-270.

48. A. HERTZ, E. TAILLARD et D. DE WERRA, Tabu Search, dans *Local Search in Combinatorial Optimization*, J. K. LENSTRA (Ed.), à venir, aussi rapport ORWP 92/18, Département de mathématiques, École Polytechnique Fédérale de Lausanne, Suisse, 1992.

49. A. HERTZ et D. DE WERRA, Using Tabu Search Techniques for Graph Coloring, *Computing*, 1987, *29*, p. 345-351.

50. A. HERTZ et D. DE WERRA, Informatique et horaires scolaires, *Output*, 1989, *12*, p. 53-56.

51. A. HERTZ et D. DE WERRA, The Tabu Search Metaheuristic: How We Used It, *Annals of Mathematics and Artificial Intelligence*, 1990, *1*, p. 111-121.

52. J. H. HOLLAND, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, MI, 1975.

53. B. JAUMARD, P. HANSEN et M. POGGI DE ARAGAO, Column Generation Methods for Probabilistic Logic, *ORSA Journal on Computing*, 1991, *3*, p. 135-148.

54. B. Jaumard, M. Stan et J. Desrosiers, Tabu Search and a Quadratic Relaxation for the Satisfiability Problem, *Cahiers du GERAD G-93-25*, GERAD, École des Hautes Études Commerciales, Montréal, 1993.

55. J. P. Kelly, B. L. Golden et A. A. Assad, Large-Scale Controlled Rounding Using Tabu Search with Strategic Oscillation, *Annals of Operations Research*, 1992, *41*, p. 69-84.

56. S. Kirkpatrick, C. D. Jr. Gelatt et M. P. Vecchi, Optimization by Simulated Annealing, *Science, 220* (4598), 1983, p. 671-680.

57. M. Laguna, J. W. Barnes et F. Glover, Tabu Search Methods for a Single Machine Scheduling Problem, *Journal of Intelligent Manufacturing*, 1991, *2*, p. 63-74.

58. M. Laguna et F. Glover, Integrating Target Analysis and Tabu Search for Improved Scheduling Systems, *Expert Systems with Applications: An International Journal*, 1992 (à paraître).

59. M. Laguna et J. L. Gonzalez-Velarde, A Search Heuristic for Just-in-Time Scheduling in Parallel Machines, *Journal of Intelligent Manufacturing*, 1991, *2*, p. 253-260.

60. S. Lin, Computer Solutions of the Traveling Salesman Problem, *Bell System Technical Journal*, 1964, *44*, p. 2245-2269.

61. M. Malek, M. Guruswamy, M. Pandya et H. Owens, Serial and Parallel Simulated Annealing and Tabu Search Algorithms for the Traveling Salesman Problem, *Annals of Operations Research*, 1989, *21*, p. 59-84.

62. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller et E. Teller, Equation of State Calculations by Fast Computing Machines, *Journal of Chemical Physics*, 1953, *21* (6), p. 1087-1091.

63. E. L. Mooney et R. L. Rardin, Tabu Search for a Class of Scheduling Problems, *Annals of Operations Research*, 1992, *41*, p. 253-278.

64. S. Oliveira et G. Stroud, A Parallel Version of Tabu Search and the Path Assignment Problem, *Heuristics for Combinatorial Optimization*, 1989, *4*, p. 1-24.

65. I. Or, *Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking*, Ph. D. Dissertation, Northwestern University, Evanston, IL, 1976.

66. I. H. Osman, Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem, *Annals of Operations Research*, 1993, *41*, p. 421-451.

67. M. Pirlot, General Local Search Heuristics in Combinatorial Optimization: A Tutorial, *Belgian Journal of Operations Research, Statistics and Computer Science*, 1992, *32*, p. 8-67.

68. J.-Y. Potvin, T. Kervahut, B. L. Garcia et J.-M. Rousseau, *A Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows*, publication CRT-855, Centre de recherche sur les transports, Université de Montréal, 1993.

69. V. M. Pureza et P. M. França, *Vehicle Routing Problems via Tabu Search Metaheuristic*, publication CRT-747, Centre de recherche sur les transports, Université de Montréal, 1991.

70. Y. Rochat et F. Semet, A Tabu Search Approach for Delivering Pet Food and Flour in Switzerland, *Journal of the Operational Research Society*, 1994, *45*, p. 1233-1246.

71. F. Semet et E. Taillard, Solving Real-Life Vehicle Routing Problems Efficiently Using Tabu Search, *Annals of Operations Research*, 1993, *41*, p. 469-488.

72. J. Skorin-Kapov, Tabu Search Applied to the Quadratic Assignment Problem, *ORSA Journal on Computing*, 1990, *2*, p. 33-45.

73. J. Skorin-Kapov, Extensions of a Tabu Search Adaptation to the Quadratic Assignment Problem, *Computers and Operations Research*, 1994, *21*, p. 855-865.

74. P. SORIANO et M. GENDREAU, Diversification Strategies in Tabu Search Algorithms for the Maximum Clique Problem, *Annals of Operations Research*,1996, *63*, p. 189-207.
75. P. SORIANO et M. GENDREAU, Tabu Search Algorithms for the Maximum Clique Problem, dans *Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge*, D. S. JOHNSON et M. A. TRICK (Ed.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 26, American Mathematical Society, 1996.
76. B. SRIVASTAVA et W.-H. CHEN, Part Type Selection Problem in Flexible Manufacturing Systems: Tabu Search Algorithms, *Annals of Operations Research*, 1993, *41*, p. 279-297.
77. M. SUN et P. G. MCKEOWN, Tabu Search Applied to the General Fixed Charge Problem, *Annals of Operations Research*, 1993, *41*, p. 405-420.
78. E. TAILLARD, *Parallel Tabu Search for the Jobshop Scheduling Problem*, rapport ORWP 89/11, Département de mathématiques, École Polytechnique Fédérale de Lausanne, Suisse, 1989.
79. E. TAILLARD, Some Efficient Heuristic Methods for the Flowshop Sequencing Problem, *European Journal of Operations Research*, 1990, *47*, p. 65-74.
80. E. TAILLARD, Robust Taboo Search for the Quadratic Assignment Problem, *Parallel Computing*, 1991, *17*, p. 443-455.
81. E. TAILLARD, Parallel Iterative Search Methods for Vehicle Routing Problems, *Networks*, 1993, *23*, p. 661-673.
82. D. DE WERRA et A. HERTZ, Tabu Search Techniques: A Tutorial and an Application to Neural Networks, *OR Spektrum*, 1989, *11*, p. 131-141.
83. M. WIDMER et A. HERTZ, A New Method for the Flow Sequencing Problem, *European Journal of Operations Research*, 1989, *41*, pp. 186-193.
84. M. WIDMER, Job Shop Scheduling with Tooling Constraints: A Tabu Search Approach, *Journal of the Operational Research Society*, 1991, *24* (1), p. 75-82.
85. J. A. G. WILLARD, *Vehicle Routing Using r-Optimal Tabu Search*, M. Sc. Dissertation, The Management School, Imperial College, Londres, 1989.
86. D. L. WOODRUFF et M. L. SPEARMAN, Sequencing and Batching for Two Classes of Jobs with Deadlines and Setup Times, *Production and Operations Management*, 1992, *1* (1), p. 87-102.
87. D. L. WOODRUFF et E. ZEMEL, Hashing Vectors for Tabu Search, *Annals of Operations Research*, 1992, *41*, p. 123-137.