Une approche de contrôle autonome pour la recherche locale

Julien Robet, Frédéric Lardeux, Frédéric Saubion

Une approche de contrôle autonome pour la recherche locale

Julien Robet Frédéric Lardeux Frédéric Saubion

LERIA, Université d'Angers
{nom}@info.univ-angers.fr

Résumé

Les algorithmes de recherche locale s'appuient principalement sur la notion de voisinage pour visiter des configurations successives au sein de l'espace de recherche, guidés également par les possibles stratégies de choix de ces voisins. Ces mouvements de proche en proche doivent être soigneusement contrôlés si l'on souhaite garantir une exploration suffisante de l'espace de recherche, ainsi que la possibilité de converger vers une solution. Des alternatives aux approches classiques ont été proposées afin d'envisager des voisinages plus complexes ou même de gérer simultanément plusieurs voisinages. Toutefois, biais souvent incontournable des métaheuristiques, ces approches induisent de nouveaux paramètres ou des heuristiques ad-hoc, dont l'importance sur les performances est grande mais le réglage délicat. Nous proposons ici, une approche de contrôle autonome pour un algorithme de recherche locale qui disposera de plusieurs opérateurs de mouvements, dont l'efficacité peut être diverse, et dont l'application respective sera ajustée en fonction de l'observation de l'état courant de la recherche, dans le but de s'adapter aux exigences d'équilibrage entre exploitation et exploration de l'espace des configurations. Nous définissons pour cela une nouvelle mesure de diversité, couplée à une mesure classique de qualité, qui autorise ainsi un contrôle simple de l'algorithme. Notre approche est finalement évaluée sur le problème classique d'affectation quadratique (QAP), obtenant des résultats prometteurs.

Abstract

Local search algorithms rely on the concept of neighborhood in order to visit successive configurations of the search space, also guided by the possible strategies for choosing these neighbors. These step by step moves must be carefully controlled if one wants to ensure a sufficient exploration of the serach space and the ability to converge towards a solution. Alternatives to conventional approaches have been proposed to consider neighborhoods or even to manage multiple neighborhoods. However, as an unavoidable bias of metaheuristics, these approaches induce new parameters or ad-hoc heuristics, whose impact on the performance is great but whose setting can be difficult. We propose here an approach for autonomous control of a local search algorithm, which has several moves operators, whose effeciency can be diverse and whose respective application is adjusted according to the observation of the current status of search, in order to adapt to the balance between exploitation and exploration of search space. To this aim, we define a new measure of diversification, coupled with a standard measure of quality, that allows us to easily control the algorithm. Our approach is finally experimented on the classical quadratic assignment problem (QAP), obtaining promising results.

1 Introduction

Les algorithmes de recherche locale [1, 23] sont des métaheuristiques qui ont été largement utilisées pour la résolution de problèmes combinatoires complexes aussi bien dans la communauté scientifique de la programmation par contraintes que dans celle de la recherche opérationnelle. Considérant un problème d'optimisation sous contraintes, ces méthodes de résolution s'appuient sur la définition d'une structure de voisinage au sein de l'espace de recherche, c'est à dire de l'ensemble des configurations possibles du problème. Une fois muni de cette relation de voisinage, l'algorithme explore alors cet espace en se déplaçant de proche en proche, guidé également par une fonction objectif qui permet d'évaluer la qualité des configurations rencontrées. Ces approches étant, par nature, incomplètes, elles permettent d'obtenir une solution sous optimale de bonne qualité, ou éventuellement une solution optimale. L'efficacité d'une telle procédure réside alors dans sa capacité à correctement explorer des zones variées de l'espace de recherche mais également dans sa propension à converger vers un optimum local,

relativement à la notion de voisinage, qui peut s'avérer, comme l'espère l'utilisateur, être également global. Le concept, particulièrement connu dans la communauté des algorithmes évolutionnaires, de balance entre intensification et diversification est un élément crucial à prendre en compte lors de la conception d'un algorithme de recherche locale. En effet, un des écueils classiques que rencontrent ces algorithmes est l'attraction excessive des minima locaux qui piègent le processus de recherche, lorsque l'ensemble des voisins potentiels sont de moins bonne qualité que la configuration courante, et que les stratégies de déplacement sont principalement fondées sur l'amélioration. Pour palier cet excès d'exploitation de l'espace de recherche (i.e., d'intensification) des mécanismes d'échappement ont alors été proposés dans un souci de diversification.

Les premières idées ont consisté à inclure, dans le choix du prochain mouvement à effectuer, des perturbations aléatoires [21], ou plus contrôlées comme dans le cas du recuit simulé [12]. Par la suite, une autre idée à consisté à utiliser d'autres fonctions (relations) de voisinages afin, par exemple, d'explorer plus largement l'espace par des voisinages de grande taille [2] ou encore, en utilisant plusieurs voisinages au sein d'un même algorithme [11, 19]. Si de telles approches permettent d'améliorer les performances des algorithmes, elles induisent par contre de nouveaux paramètres et/ou heuristiques qu'il convient de gérer correctement, ce qui constitue une tâche fastidieuse pour l'utilisateur. En effet, le réglage des paramètres reste un problème épineux pour le concepteur et l'utilisateur de telles métaheuristiques. Si l'ultime but de cette quête est de trouver des réglages universels permettant de traiter de larges classes de problèmes, l'utilisateur recourt, la plupart du temps, à une série d'expériences ainsi qu'à sa connaissance du problème pour tenter de formuler des règles empiriques d'ajustement de ces paramètres. On peut citer ici l'étude proposée par B. Mazure et al. [18], dans laquelle une valeur optimale de longueur de liste taboue est proposée pour une classe d'instances du problème SAT. Toutefois, il semble, de nos jours, préférable de s'orienter vers des algorithmes plus autonomes, incluant des mécanismes leur permettant de prendre en charge eux-même le réglage de leurs paramètres et de leurs heuristiques. Une telle approche nécessite alors l'intégration d'outils et techniques issues d'autres domaines de l'informatique, notamment de l'apprentissage artificiel (nous renvoyons le lecteur à [4, 10] pour plus de détails). Ce type d'approches de contrôle peut être schématisé par la figure 1.

Dans ce contexte, et inspirés par les travaux précédents que nous avons menés sur la gestion autonomes d'opérateurs multiples dans les algorithmes génétiques [15, 17], nous proposons ici une nouvelle approche pour concevoir un algorithme de recherche locale incluant plusieurs opérateurs de mouvements, correspondant à des voisinages et des stratégies de déplacement différents, dont le contrôle sera pris en charge automatiquement. Notre but est de montrer que nous pouvons bénéficier des atouts respectifs de ces opérateurs et décider de leur utilisation en fonction de la situation courante de la recherche. Pour cela, et afin d'ajuster convenablement la balance entre intensification et diversification, nous définissons un nouveau critère d'évaluation de la diversité pour la recherche locale qui sera couplé au critère, plus classique, de qualité, lié à la fonction objectif et aux contraintes du problème à résoudre. Nous utilisons ensuite ces mesures pour décider du prochain mouvement à effectuer. Afin d'évaluer les bénéfices de notre approche, nous avons expérimenté notre algorithme sur le fameux problème de l'affectation quadratique (QAP) qui a été largement étudié et pour lequel nous disposons d'une imposante libraire d'instances et de résultats [5].

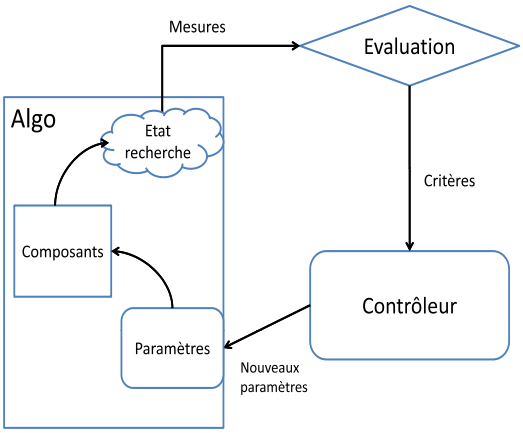

FIG. 1 – Schéma général

Dans la section 2 nous effectuons quelques rappels sur les principes de base de la recherche locale ainsi que sur les méthodes destinées à rendre la recherche locale plus autonome et adaptative. Dans la section 3, nous proposons notre nouvelle mesure de diversité pour l'étude des performances de la recherche locale. La section 4 présente le problème d'affectation quadratique sur lequel nous avons expérimenté notre approche ainsi que les opérateurs de mouvements que nous avons utilisés pour notre recherche locale. La section 5 présente notre algorithme ALS et enfin, la section 6 est consacrée à l'expérimentation de l'approche.

2 Vers une recherche locale autonome

Nous nous plaçons ici dans le contexte de la résolution de problèmes d'optimisation combinatoires classiquement définis par un ensemble de variables $Var = \{x_1, \cdots, x_n\}$, auxquelles sont associés des domaines de valeurs possibles $D = \{D_1, \cdots, D_n\}$, D_i étant l'ensemble des valeurs que peut prendre la variable x_i. L'espace de recherche S correspond au produit cartésien des domaines, $S = \Pi_{i=1}^{n} D_i$. Nous considérons un ensemble de contraintes C et une solution réalisable est alors une affectation de valeurs aux variables satisfaisant toutes les contraintes de C et respectant les domaines initiaux de D. Nous considérons également une fonction objectif $f : S \rightarrow \mathbb{R}$. Une solution optimale est une solution réalisable maximisant ou minimisant, selon le cas, la fonction f.

En programmation par contraintes, diverses approches sont envisageables pour résoudre de tels problèmes. On distingue classiquement les approches complètes, basées sur des parcours arborescents de l'espace de recherche couplés à des techniques d'élagage, et les approches incomplètes, principalement basées sur des méthodes métaheuristiques. L'exploration systématique de l'espace de recherche étant exclue, les métaheuristiques proposent des schémas généraux permettant de restreindre cette exploration à certaines zones. D'une manière très générale, deux stratégies fondamentales d'exploration et d'exploitation peuvent être identifiées :

- l'intensification, dont le but est de chercher une solution dans une zone réduite de l'espace,
- la diversification, qui permet de se déplacer d'une zone vers une autre zone, éventuellement lointaine, et autorise ainsi un balayage large de l'espace de recherche.

Ces notions sont mises en œuvre au sein des différentes métaheuristiques et l'efficacité relative d'une méthode réside essentiellement dans la gestion de leur complémentarité. La coordination de ces stratégies vise à éviter les problèmes induits par la présence d'optima locaux propres à la structure du problème traité et inhérents à ce type d'approche. Par exemple, dans le cas d'un algorithme de recherche par voisinage, une stratégie d'amélioration systématique de la configuration courante conduira le processus de recherche vers un optimum local où il risque de stagner.

Nous nous concentrons ici sur les méthodes de recherche locale (dites aussi par voisinage) [1, 23]. L'exploration se fait parmi les points voisins du point courant en s'appuyant sur une fonction qui estime leur qualité potentielle. Étant donné l'espace de recherche S, on se donne une fonction de voisinage $\mathcal{N} : S \rightarrow 2^S$ et une fonction d'évaluation $eval : S \rightarrow \mathbb{R}$, qui prend en compte la satisfaction des contraintes C du problème ainsi que la fonction objectif f. On définit deux actions élémentaires :

- l'amélioration qui consiste, à partir d'une configuration c, à choisir une configuration voisine c' dont la valeur $eval(c')$ améliore $eval(c)$ (cette notion varie selon que l'on veut maximiser ou minimiser le critère d'évaluation).
- le déplacement arbitraire qui consiste à choisir n'importe quelle configuration voisine c'.

L'efficacité de la recherche locale est alors liée à la gestion de la balance entre intensification et diversification. Cette gestion peut s'opérer, au sein de l'algorithme de recherche locale, par le biais de paramètres comportementaux, qui permettent d'ajuster l'utilisation des composants de l'algorithme (par exemple, la probabilité d'effectuer un mouvement aléatoire ou bien la longueur de la liste taboue), ou de paramètres structurels, qui sont directement liés aux composants eux-mêmes de l'algorithme (par exemple, le choix d'une fonction de voisinage, la fonction d'évaluation ou bien le choix d'une stratégie d'amélioration pour un processus de descente). Nous allons donc brièvement aborder ces deux aspects dans l'optique de la recherche locale autonome et nous recommandons au lecteur le récent ouvrage de R. Battiti et al. [4], pour plus de précisions.

2.1 Réglage de paramètres pour la recherche locale

Comme pour tout algorithme de type métaheuristique, le réglage des paramètres reste l'une des contraintes essentielles de leur utilisation pratique. Le réglage de paramètres a largement été étudié dans la communauté des algorithmes évolutionnaires [14]. Comme mentionné en introduction, cette tâche s'avère particulièrement ardue et repose sur le savoir-faire, bien souvent empirique, de l'utilisateur et sur ses connaissances des caractéristiques du problème à résoudre. Dès lors, l'ajustement de paramètres s'appuie sur une série, en général coûteuse, d'expériences. Une telle approche réduit considérablement l'universalité des valeurs obtenues et la transposition de ces expérimentations à d'autres problèmes. Une autre voie consiste alors à envisager un contrôle des paramètres durant l'exécution de l'algorithme. Ce contrôle peut être supervisé par un ensemble de connaissances acquises au préalable ou encore être abordé dans une optique plus autonome, les paramètres évoluant en fonction de l'état courant de la recherche. Le contrôle autonome de paramètres pour la recherche locale est largement illustré dans l'approche de recherche réactive [4]. De telles approches se développent à la frontière d'autres domaines de l'informatique, faisant notamment appel à des techniques issues de l'apprentissage

automatique ou de la programmation génétique.

En dehors des paramètres classiques et du point de vue des composants de l'algorithme, on peut également s'intéresser à faire évoluer la fonction d'évaluation en fonction de l'état de la recherche avec des systèmes de pénalités (voir également [4]) ou encore apprendre des stratégie adéquates de manière automatique [9]. Ici nous nous concentrons sur les aspects liés à la notion de voisinage.

2.2 Voisinages larges et multiples

Comme mentionné précédemment, le principe fondateur de la recherche locale est de parcourir l'espace de recherche selon une relation de voisinage définie entre les configurations possibles du problème. Le voisinage détermine ainsi la notion de structure de l'espace de recherche, on parle de paysage de recherche, dont l'importance est cruciale vis à vis des notions d'optima locaux ou globaux, évoquées plus haut. Dès lors, il parait naturel de s'intéresser à ce voisinage comme un moyen de gérer la balance entre intensification et diversification. On peut par exemple, pour diversifier la recherche, envisager d'utiliser des voisinages de tailles différentes, permettant de varier les possibilités de mouvements [11]. Il est également possible d'envisager des voisinages très larges [2]. Bien évidemment, l'extension du voisinage a un impact important sur le temps nécessaire à son évaluation pour décider du prochain mouvement.

2.3 Vers une approche englobante

Dans cet article, notre idée est de combiner finalement ces deux aspects (paramètres et composants) en considérant plutôt des opérateurs de mouvements en réglant automatiquement leur application au cours de la recherche. Nous introduisons donc, au sein de la recherche locale, une méthode de sélection d'opérateur adaptative, telle que nous l'avons étudiée dans [15] pour les algorithmes génétiques. L'objectif est donc d'évaluer, après application, l'impact des opérateurs et de se servir de cette évaluation pour ajuster leur utilisation au cours de la recherche. Cette approche peut être résumée par la figure 2.

Cette figure nous permet de mettre en lumière les principales questions liées au contrôle d'un algorithme de recherche locale :

- Comment évaluer la situation en cours ?
- Comment attribuer des récompenses aux opérateurs en fonction de cette évaluation ?
- Comment utiliser ces récompenses pour sélectionner l'opérateur à appliquer pour faire le prochain pas de recherche locale ?

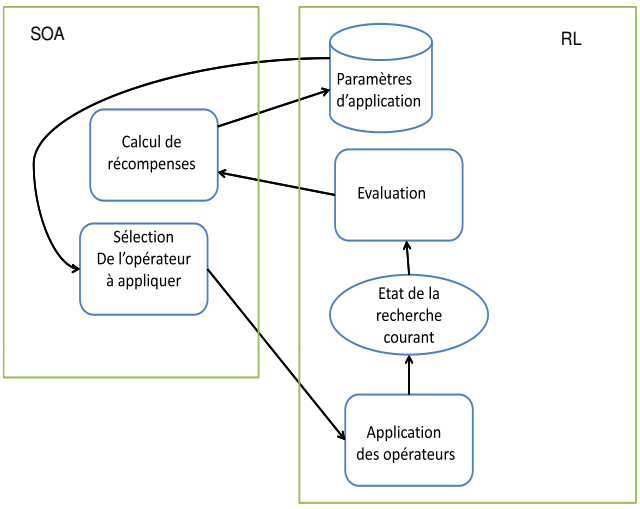

FIG. 2 – Recherche locale autonome

Alors que nous pouvons nous inspirer des mécanismes existants pour les algorithmes évolutionnaires [15] en ce qui concerne les deux derniers points, la question de l'évaluation est plus délicate. En effet, alors que dans le cas des algorithmes évolutionnaires, les critères de qualité et d'entropie de la population ont été étudiés [16, 17], il n'en est pas de même pour la recherche locale. En effet, si la notion de qualité permet également de guider et d'évaluer la recherche en cours, la notion de diversité mérite plus d'attention. Pourtant, il est apparu, dans nos travaux précédents [16, 17], que cette notion est très largement complémentaire de la qualité si l'on veut correctement gérer la balance entre diversification et intensification. Dans la section suivante, nos allons donc nous pencher sur la définition d'une nouvelle mesure de diversité adaptée à la recherche locale.

3 Une mesure de diversité pour la recherche locale

Le concept de diversité est souvent employé pour caractériser un parcours de recherche locale. Il est assez intuitif et pourrait se résumer ainsi : un parcours est d'autant plus diversifié qu'il a visité des régions éloignées de l'espace de recherche. La diversification est un élément de première importance pour un système de résolution dans le sens où elle évite de focaliser la fouille de l'espace de recherche dans des zones confinées, et tend à orienter le système vers des régions attractives.

Cette notion, bien que couramment évoquée par les concepteurs de métaheuristiques, est difficilement calculable en pratique. En effet, une telle mesure dépend de plusieurs éléments comme, par exemple, la façon dont est codée une configuration, la taille de l'espace de recherche. Le cas des méthodes de type évolutionnaire doit cependant être traité séparément. En effet, un algorithme manipulant une population de configurations peut évaluer la diversité de la recherche en comparant les différents individus composant la population à chaque itération. En général, ce procédé consiste à évaluer une distance moyenne entre les individus, par exemple à l'aide d'un calcul d'entropie. Le lecteur intéressé trouvera des informations à ce sujet dans [6].

Les méthodes à base de recherche locale pure ne peuvent pas bénéficier de ces techniques, du fait qu'elles ne traitent qu'une seule configuration à chaque pas de la recherche. La diversité de cette dernière doit donc être calculée en fonction du parcours effectué.

3.1 Pré-requis pour une mesure pertinente

Malgré le succès et l'étude poussée des méthodes de recherche locale durant ces dernières années, peu de travaux ont été menés dans le but de quantifier la diversité de tels procédés. A. Linhares introduit dans [13] les notions relatives au sujet, définit les caractéristiques que doit respecter une telle mesure, et propose une méthode d'évaluation de la diversité basée sur les mesures de distance entre configurations. Nous nous efforçons donc de respecter les contraintes suivantes pour notre système d'évaluation de diversité qui se présentera comme une fonction div :

- son domaine sera le parcours P assimilé à l'ensemble des configurations qui le composent.
- son co-domaine sera l'intervalle fermé $[0, 1]$. En effet, cette normalisation permet d'obtenir des valeurs précises, indépendantes de la taille du problème et de celle du parcours.
- $div(P) = 0$ ssi $|P| = 0$

3.2 Vers une nouvelle mesure

Afin de quantifier la diversité d'un parcours de recherche locale, la mesure que nous proposons consiste à analyser, pour chaque variable décisionnelle, la répartition des valeurs affectées dans son domaine respectif. Notre approche part en effet d'un constat simple : dans le cas où un parcours de recherche est très diversifié, les valeurs successives affectées aux différentes variables auront tendance à "couvrir" leur domaine. Dans le cas contraire, un parcours peu diversifié sera caractérisé par l'affectation d'un faible nombre de valeurs différentes à chaque variable (relativement au cardinal de son domaine), spécialement pour celles dont la valeur reste inchangée pendant le parcours étudié. L'évaluation consiste donc à observer, pour chaque variable décisionnelle, l'écart type du nombre d'occurrences de chaque valeur possible pour cette variable (i.e., pour chaque élément de son domaine). La moyenne de ces écarts type traduit la similitude du parcours étudié. En effet, les écarts types calculés seront d'autant plus faibles que le parcours étudié sera diversifié. Afin d'être normalisée, la mesure est ensuite divisée par la similitude maximale théoriquement atteignable. On obtient donc une valeur comprise entre 0 et 1 que l'on va retrancher de 1 pour traduire la diversité du parcours.

Plus formellement, étant donné un espace de recherche $S, P \subseteq S$ est un parcours de recherche locale constitué d'un ensemble de $|P|$ configurations. Soit $x \in Var$ une des variables décisionnelles du problème traité. On définit alors $aff(x, c)$ la valeur que prend la variable x pour la configuration $c \in P$ ($aff(x, c) \in D_x$).

Nous nous intéressons au nombre d'affectations donnant la valeur v à la variable x au cours du parcours P, que nous définissons comme :

$$nbutil(x, v, P) = |\{c \in P | aff(x, c) = v\}|$$

L'écart type du nombre d'occurrences des différentes valeurs prises par la variable x est donné par :

$$et(x, P) = \sqrt{\frac{\Sigma_{v \in D_x} \left(nbutil(x, v, P) - (\frac{|P|}{|D_x|})\right)^2}{|D_x|}}$$

L'écart type maximal atteignable lors d'un parcours P pour la variable x (cas où une seule valeur est utilisée pour chaque configuration de P) est alors :

$$etmax(x, P) = \sqrt{\frac{(|P| - 1).(\frac{|P|}{|D_x|})^2 + (|P| - \frac{|P|}{|D_x|})^2}{|D_x|}}$$

La similitude des configurations qui composent le parcours P étudié peut donc maintenant être définie par :

$$sim(P) = \frac{\Sigma_{x \in Var} et(x, P)}{|Var|}$$

La similitude maximale que l'on peut obtenir avec un parcours P (où toutes les configurations sont identiques) est :

$$simmax(P) = \frac{\Sigma_{x \in Var} etmax(x, P)}{|Var|}$$

La diversité est alors définie comme :

$$div(P) = \begin{cases} 0 \text{ si } |P| = 0 \\ 1 - \frac{sim(P)}{simmax(|P|)} \text{ sinon} \end{cases}$$

Deux cas extrême peuvent être identifiés :

- Un parcours P_1 a une diversité minimale si toutes les configurations de P_1 sont égales, c'est-à-dire que chaque variable décisionnelle x prend une valeur $|P_1|$ fois, et que les $(|D_x| - 1)$ autres valeurs ne sont jamais utilisées.
- Un parcours P_2 a une diversité maximale ssi $\forall x \in Var, \nexists (v_1, v_2) \in D_x^2, nbutil(x, v_1, P_2) - nbutil(x, v_2, P_2) > 1$. Un tel niveau n'est pour ainsi dire jamais atteint en pratique. On peut par exemple l'obtenir en affectant à chaque variable la valeur de son domaine qu'elle a le moins utilisée, et ceci à chaque itération de la recherche.

4 Application au problème d'affectation quadratique

4.1 Présentation du problème et travaux existants

Nous avons choisi de valider notre approche sur le célèbre problème de l'affectation quadratique (quadratic assignment problem, QAP), qui a été largement étudié au cours des cinquante dernières années. Intuitivement, le problème consiste à affecter des ressources à différents sites, en prenant en compte les distances qui séparent ces sites ainsi que l'importance des flux de données échangés entre les différentes ressources. Plus formellement, étant donnés n ressources et n sites, on dispose de deux matrices de taille $n \times n$, $D = [d_{i,j}]$ et $F = [f_{k,l}]$, où $d_{i,j}$ la distance entre les sites i et j et $f_{k,l}$ est la quantité d'informations transitant entre les ressources k et l. On va alors chercher une affectation π (i.e., une permutation de $1..n$) des ressources aux sites de coût $f(\pi)$ minimal. Le coût d'une telle affectation étant calculé comme suit :

$$f(\pi) = \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} d_{i,j} \cdot f_{\pi(i),\pi(j)}$$

Le problème a été démontré NP-difficile [20], et est parfois qualifié du plus difficile de cette classe de complexité, du fait que de nombreux problèmes, tels que ceux du voyageur de commerce, de la clique maximale ou de partitionnement de graphe, sont un cas particulier d'affectation quadratique. D'autre part, il présente une forte importance pratique due à sa capacité à modéliser de nombreuses situations issues du domaine industriel. Cependant, malgré la quantité de travaux qu'a suscité le problème, même les méthodes les plus performantes ne peuvent résoudre à l'optimum des instances de taille supérieur à 30. Parmi ces dernières, on citera notamment deux versions de recherche taboue. La première, nommée Robust taboo search [22], a été proposée par Taillard en 1991 qui reste à ce jour une des plus compétitive. La seconde, qui vise à régler la taille de la liste taboue automatiquement via une détection de cycles performante, a été mise au point par Battiti et Tecchiolli en 1994 sous le nom de Reactive tabu search [3]. Fleurent et Ferland ont également obtenu d'excellents résultats avec un algorithme de type génétique [8]. Enfin, notons que la méthode de recuit simulé ne semble pas vraiment adaptée au problème puisque la meilleure version de ce type d'algorithme appliquée au QAP, présentée par Connolly en 1990 [7], ne permet pas de rivaliser avec les approches évoquées plus haut. Notons par ailleurs que les résultats de chacune d'elle dépendent fortement du type d'instance sur laquelle elle est appliquée, de telle sorte qu'il est inapproprié de distinguer une d'entre elles comme supérieure aux autres. Enfin, signalons que le QAP dispose d'une librairie fournie et variée d'instances de problèmes [5] et de résultats, permettant d'évaluer les apports de notre mécanisme. Pour une étude plus approfondie du problème, nous renvoyons le lecteur intéressé au site `http ://www.seas.upenn.edu/qaplib/`.

En ce qui concerne les approches de résolution par recherche locale, il est évident que, étant donnée une permutation π correspondant à une affectation de ressources aux sites, un voisinage basique consiste à échanger deux ressources, puisque l'on souhaite respecter la contraintes d'affectation bijective et retrouver une permutation. Ainsi, on peut ensuite définir des voisinages plus larges autorisant plus d'échanges entre les sites. Nous allons donc nous servir de ces voisinages pour définir nos opérateurs de mouvements.

4.2 Opérateurs candidats

Rappelons ici que le but du présent travail n'est pas d'obtenir des résultats concurrentiels pour le QAP mais de mettre en place un contrôleur capable de tirer le meilleur profit d'un ensemble d'opérateurs aux caractéristiques a priori inconnues. Il est donc intéressant de proposer au contrôleur des opérateurs volontairement variés pour que le système puisse mettre en avant une éventuelle complémentarité. Ainsi, nous avons défini un jeu de dix opérateurs, dont les cinq premiers paraissent naturellement voués à l'intensification et cinq autres présentant une tendance à la diversification.

O1 consiste à échanger deux ressources de telle sorte que chacune se retrouve placée sur le site précédemment occupé par l'autre. L'ensemble des paires candidates à un tel échange (qui sont au nombre de $\frac{n.(n-1)}{2}$) sont examinées et la meilleure

(en terme de gain pour la fonction d'évaluation) est sélectionnée.

O2 est similaire à O1, mais ne nécessite pas systématiquement l'exploration de l'ensemble du voisinage car la première paire améliorante (qui présente un gain strictement positif) est choisie.

O3 fonctionne comme O1, mais n'opte pas obligatoirement pour la meilleure paire. A la place, il en choisit aléatoirement une parmi le sous-ensemble des k meilleures paires rencontrées. Nous avons fixé $k = 5$ pour nos expérimentations.

O4 reprend le principe de O1, mais avec une profondeur de 2 au lieu de 1, c'est-à-dire que les deux meilleures paires indépendantes p_1 et p_2 sont échangées simultanément. Par indépendantes, on entend qu'une unité ne peut appartenir à p_1 et p_2 à la fois.

O5 est identique à O4 mais avec une profondeur de 3.

O6 effectue des échanges entre 3 ressources. L'échange est obligatoirement complet, c'est-à-dire que les trois ressources impliquées dans l'échange doivent être déplacées. On a donc ici deux choix possibles, parmi lesquels celui apportant le meilleur gain est choisi. Les 3 ressources impliquées sont sélectionnées aléatoirement.

O7 est identique à O6, mais avec des échanges entre 4 ressources.

O8 est identique à O6, mais avec des échanges entre 5 ressources.

O9 est identique à O6, mais avec des échanges entre 6 ressources.

O10 effectue une succession de k' (fixé à 3 pour les expérimentations) échanges de 2 ressources choisies aléatoirement.

5 L'algorithme ALS

Le but de notre système, nommé ALS (Autonomous Local Search) est de gérer un ensemble d'opérateurs de recherche locale afin de les appliquer au moment où ils peuvent le plus améliorer la recherche. Le défit de ce procédé est donc de faire cohabiter les trois principaux modules que sont l'évaluation de l'état courant de la recherche, l'attribution de récompenses aux composants internes, et l'utilisation de ces récompenses pour choisir l'opérateur à appliquer.

5.1 Évaluation des opérateurs

Le procédé utilisé pour analyser les composants internes est largement inspiré de nos précédents travaux dans le domaine de la recherche évolutionnaire [16]. Son principe est de maintenir tout au long de la recherche un historique des performances récentes de chaque opérateur. L'originalité de la méthode vient du fait que son analyse ne se contente pas d'un seul critère de jugement (les variations de qualité apportées par un opérateur), mais examine également les écarts en diversité. C'est ici que la mesure de diversité que nous proposons section 3.2 prend toute son importance. Formellement, étant donné un opérateur on définit :

- $\Delta Q_{op,t}$ la variation moyenne de qualité résultant des t applications de *op*
- $\Delta D_{op,t}$ la variation moyenne de diversité résultant des t applications de *op*

Le paramètre t correspond à la taille de la fenêtre glissante qui stocke les informations relatives à chaque opérateur. Étant donné que nous cherchons à faire ressortir les caractéristiques des composants à un moment précis de la recherche, il est important de régler ce paramètre correctement. S'il prend une valeur trop élevée, alors l'évaluation risque de ne pas refléter les spécificités actuelles. À l'inverse, une valeur trop faible ne permettra pas de mémoriser les cas où un opérateur devient très efficace sur une courte période. La variation de qualité apportée par l'application de *op*, traditionnellement définie par $\Delta Q = eval(op(c)) - eval(c)$ a été redéfinie afin de prendre en compte le fait qu'une configuration médiocre est plus facilement améliorable qu'une autre de bonne qualité. On a alors, après pondération :

$$\Delta Q = \frac{eval(op(c)) - eval(c)}{eval(c) + 1}$$

Enfin, les écarts de diversité entre deux itérations sont calculés ainsi :

$$\Delta D = div(P_{i,j}) - div(P_{i-1,j-1})$$

où $P_{i,j}$ est l'ensemble des configurations visitées de l'itération i à l'itération j.

À partir de cela, nous mettons en place le système de récompenses suivant :

$$score(op) = \alpha.\Delta Q_{op,t} + (1 - \alpha).\Delta D_{op,t}$$

où $\alpha \in [0..1]$ et désigne l'orientation que doit adopter la recherche. Une forte (respectivement faible) valeur favorise l'intensification (respectivement la diversification). Les différentes probabilités d'application sont ensuite déduites de ces récompenses :

$$p(op_k) = p_{min} + max\Big(0, (1 - p_{min}).\frac{score(op_k)}{\sum_{i=1}^{nbop} score(op_i)}\Big)$$

La probabilité $1 \leq p_{min} \leq \frac{1}{nbOp}$ ($nbop$ représentant le nombre total d'opérateurs) assure à l'ensemble des opérateurs une chance d'être élu à chaque pas de la recherche dans le but de garantir une diversité minimale dans les sélections effectuées.

5.2 Évaluation de la recherche

En appliquant les mêmes principes de mémorisation aux fluctuations de la recherche, on obtient de précieuses informations quant à l'évolution dans l'espace de recherche. En particulier, il est utile de s'intéresser aux écarts de diversité apportés par les dernières itérations. Une perte en diversité traduit la fouille d'une zone précise de l'espace de recherche, alors qu'un gain apparaît lors de l'éloignement d'une certaine région.

5.3 Stratégies d'application

Au cours de la résolution, le choix de l'opérateur à appliquer est effectué selon les probabilités évoquées dans la section 5.1. Comme nous l'avons vu précédemment, ces probabilités sont largement influencées par un paramètre α qui modélise la balance souhaitée entre intensification et diversification. La stratégie d'application peut donc être vue comme le moyen de calculer la valeur de α en fonction des observations collectées sur l'état de la recherche. Par exemple, si l'on détecte un blocage dans un optima local, il est bénéfique de faire décroître α afin de favoriser la diversification du parcours. Ainsi, à l'heure actuelle, nous avons doté ALS d'une stratégie simpliste ($alpha \in \{0, 1\}$ au lieu de $\alpha \in [0..1]$) afin, dans un premier temps, de vérifier l'impact de la méthode : il s'agit de définir la valeur par défaut de α à 1 (intensification pure) et de ne procéder à une diversification que lorsque l'on en détecte le besoin. Ce besoin est détecté par une perte de diversité, couplée à une absence de gains en qualité, à la suite de l'application d'un opérateur. On fixe alors α à 0 pour l'itération suivante, dans le but d'effectuer un pas de diversification.

6 Résultats expérimentaux

Nous avons expérimenté ALS sur 38 instances issues de la QAPLIB [5]. Une première variante, disposant des 10 opérateurs présentés dans la section 4.2, et une seconde ne manipulant que 2 opérateurs (nommés O1 et O9 dans la section 4.2, respectivement meilleurs intensificateur et diversificateur lors des tests avec ALS-10) ont été comparées à un choix uniforme parmi les 10 opérateurs. Le tableau 1 représente les résultats obtenus par les 3 variantes, et indique à titre indicatif ceux de l'algorithme Robust Taboo Search, présenté en section 4.1. Pour chaque instance, la meilleure valeur connue (mars 2009) pour la fonction objectif est indiquée dans la colonne BKV (Best Known Value). Les résultats indiqués représentent la déviation moyenne θ_{avg} par rapport à BKV qui est calculée selon la formule :

$$\theta_{avg} = \frac{100(f_{avg} - BKV)}{BKV}$$

où f_{avg} représente la valeur moyenne de la fonction objectif sur 10 exécutions, chacune d'elles ayant duré un nombre de secondes spécifié dans la colonne temps.

Les résultats du tableau 1 font clairement ressortir l'intérêt du contrôleur puisque sur les 38 instances testées, seulement six ont apporté de meilleurs résultats avec un choix uniforme. De plus, cinq d'entre elles appartiennent à la famille bur26 où ALS obtient des résultats très semblables. Pour les instances structurées, comme sko* ou tai*a, notre approche semble avoir un très bon comportement (le choix uniforme n'obtient qu'une seule fois de meilleurs résultats). Notons également que l'élargissement de la gamme d'opérateurs manipulés par le système est bénéfique pour l'efficacité de la résolution. En effet, ALS-10 a été plus performant qu'ALS-2 pour 31 des 38 instances testées.

7 Conclusion

Ce travail pose les bases du socle sur lequel nous comptons nous appuyer pour développer diverses stratégies, et mener nombre d'expérimentations. En effet, les différents composants impliqués nécessitent une étude plus poussée afin d'accroître la qualité de leur contribution au système. Notamment, le module chargé de maintenir les besoins en intensification et en diversification, un élément crucial de la démarche, est loin d'avoir atteint un comportement optimal. Des recherches complémentaires à ce sujet devraient significativement améliorer l'approche. Une seconde piste intéressante réside dans l'élargissement de la gamme des opérateurs candidats que manipule le contrôleur. A l'heure actuelle, les dix opérateurs utilisés sont plutôt simples et ont été définis *à la main*. L'étape suivante consistera à définir de nouveaux opérateurs reproduisant, sur un nombre limité d'itérations, le comportement des méthodes les plus efficaces pour le problème étudié (en l'occurrence celui de l'affectation quadratique). Nous prévoyons également de mettre en place un procédé visant à générer automatiquement de nouveaux opérateurs. Le contrôleur prendrait alors une fonction double puisqu'en plus d'organiser la séquence d'applications des différents opérateurs, il permettrait d'évaluer les caractéristiques de ceux régulièrement générés, et ainsi conserver les plus prometteurs d'entre eux.

Instance	BKV	temps	ALS 10 op	ALS 2 op	CU 10 op	Ro-TS
bur26a	5426670	50	0,0701	**0,0698**	0,1177	0,0004
bur26b	3817852	50	0,1244	0,1274	**0,1084**	0,0032
bur26c	5426795	50	**0,0236**	0,0266	0,0359	0,0004
bur26d	3821225	50	**0,0205**	0,0218	0,0214	0,0015
bur26e	5386879	50	0,0462	0,1260	**0,0366**	0
bur26f	3782044	50	0,1360	0,1019	**0,0153**	0,0007
bur26g	1,0E+07	50	0,1239	0,1621	**0,0251**	0,0003
bur26h	7098658	50	0,0622	0,2453	**0,0164**	0,0027
chr25a	3796	40	18,3878	**15,9484**	42,4341	6,9652
els19	1,7E+07	20	**1,3656**	7,4253	4,8532	0
kra30a	88900	76	**1,0146**	1,6097	3,8774	0,4702
kra30b	91420	86	**0,0831**	0,2111	2,4251	0,0591
tai20b	1,2E+08	27	**0,1399**	0,1810	1,6870	0
tai25b	3,4E+08	50	**0,1891**	4,2186	1,7558	0,0072
tai30b	6,4E+08	90	**0,8338**	4,1695	1,5794	0,0547
tai35b	2,8E+08	147	2,2365	2,4788	**1,4712**	0,1777
tai40b	6,4E+08	240	**1,5322**	3,7357	2,0287	0,2082
tai50b	4,6E+08	480	1,5832	**0,9594**	1,4307	0,2943
tai60b	6,1E+08	855	**0,5568**	0,7787	1,4344	0,3904
tai80b	8,2E+08	2073	**1,1055**	1,5086	1,5163	1,4354
nug20	2570	30	**0,2802**	0,5058	1,9767	0,101
nug30	6124	83	**0,2743**	0,3919	2,0901	0,271
sko42	15812	248	**0,1505**	0,2416	2,0529	0,187
sko49	23386	415	**0,3464**	0,3831	2,0174	0,198
sko56	34458	639	**0,2969**	0,3419	2,1139	0,347
sko64	48498	974	**0,3784**	0,4555	2,1989	0,221
sko72	66256	1415	**0,5162**	0,5687	2,4864	0,478
sko81	90998	2041	4,1135	**4,0523**	5,3981	0,304
sko90	115534	2825	0,4368	**0,4130**	2,2731	0,386
tai20a	703482	26	2,0103	**1,5700**	3,6558	0,769
tai25a	1167256	50	**1,8540**	1,8858	3,8223	1,128
tai30a	1818146	87	1,8407	**1,6269**	3,6971	0,871
tai35a	2422002	145	**1,6184**	1,8995	3,9348	1,356
tai40a	3139370	224	**1,6309**	2,0021	4,0596	1,284
tai50a	4941410	467	**1,6362**	2,0849	4,4437	1,377
tai60a	7208572	820	**1,5267**	2,1955	4,7171	1,544
tai80a	1,4E+07	2045	**1,0563**	2,3685	4,7724	1,170
wil50	48816	441	**0,0639**	0,1135	1,0005	0,137

TAB. 1 – Déviation moyenne d'ALS (avec 2 et 10 opérateurs) et d'un choix uniforme parmi les 10 opérateurs par rapport aux meilleurs résultats connus (BKV)

Références

[1] Emile Aarts and Jan Karel Lenstra, editors. *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[2] Ravindra K. Ahuja, Krishna C. Jha, Krishna C. Jha, James B. Orlin, James B. Orlin, Dushyant Sharma, and Dushyant Sharma. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123 :75–102, 2002.

[3] R. Battiti and Tecchiolli. The reactive tabu search. *ORSA Journal on Computing*, 6(2) :126–140, 1994.

[4] Roberto Battiti, Mauro Brunato, and Franco Mascia. *Reactive Search and Intelligent Optimization*, volume 45 of *Operations research/Computer*

Science Interfaces. Springer Verlag, 2008.

[5] R. E. Burkard, S. Karisch, and F. Rendl. Qaplib-a quadratic assignment problem library. *European Journal of Operational Research*, 55(1) :115–119, November 1991.

[6] Edmund K. Burke, Steven M. Gustafson, Graham Kendall, and Natalio Krasnogor. Advanced population diversity measures in genetic programming. In *Parallel Problem Solving from Nature - PPSN VII, 7th International Conference*, volume 2439 of *Lecture Notes in Computer Science*, pages 341–350. Springer, 2002.

[7] David T. Connolly. An improved annealing scheme for the qap. *European Journal of Operational Research*, 46(1) :93–100, May 1990.

[8] Charles Fleurent, Jacques, and A. Ferland. Genetic hybrids for the quadratic assignment problem. In *DIMACS Series in Mathematics and Theoretical Computer Science*, pages 173–187. American Mathematical Society, 1994.

[9] A. Fukunaga. Automated discovery of local search heuristics for satisfiability testing. *Evolutionary Computation*, 16(1) :31–61, 2008.

[10] Youssef Hamadi, Eric Monfroy, and Frederic Saubion. Special issue on autonomous searcch. *Contraint Programming Letters*, 4, 2008.

[11] Pierre Hansen and Nenad Mladenovi. A tutorial on variable neighborhood search. Technical report, Les Cahiers du GERAD, HEC Montreal and GERAD, 2003.

[12] Scott Kirkpatrick, D. Gelatt Jr., and Mario P. Vecchi. Optimization by simmulated annealing. *Science*, 220(4598) :671–680, 1983.

[13] Alexandre Linhares. The structure of local search diversity. In *Math'04 : Proceedings of the 5th WSEAS International Conference on Applied Mathematics*, pages 1–5, Stevens Point, Wisconsin, USA, 2004. World Scientific and Engineering Academy and Society (WSEAS).

[14] Fernando G. Lobo, Cláudio F. Lima, and Zbigniew Michalewicz, editors. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*. Springer, 2007.

[15] J. Maturana, A. Fialho, F. Saubion, M. Schoenauer, and M. Sebag. Compass and dynamic multi-armed bandits for adaptive operator selection. In *Proceedings of IEEE Congress on Evolutionary Computation CEC*, 2009. to appear.

[16] J. Maturana and F. Saubion. Towards a generic control strategy for EAs : an adaptive fuzzy-learning approach. In *Proceedings of IEEE International Conference on Evolutionary Computation (CEC)*, pages 4546–4553, 2007.

[17] Jorge Maturana and Frederic Saubion. A compass to guide genetic algorithms. In G. Rudolph et al., editor, *Proc. PPSN'08*, pages 256–265. Springer, 2008.

[18] Bertrand Mazure, Lakhdar Sais, and Éric Grégoire. Tabu search for sat. In *AAAI/IAAI*, pages 281–285, 1997.

[19] Nenad Mladenovic and Pierre Hansen. Variable neighborhood search. *Computers & OR*, 24(11) :1097–1100, 1997.

[20] Sartaj Sahni and Teofilo F. Gonzalez. P-complete approximation problems. *J. ACM*, 23(3) :555–565, 1976.

[21] Bart Selman, Henry A. Kautz, and Bram Cohen. Noise strategies for improving local search. In *AAAI*, pages 337–343, 1994.

[22] Éric D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17(4-5) :443–455, 1991.

[23] Pascal Van Hentenryck and Laurent Michel. *Constraint-Based Local Search*. The MIT Press, 2005.