



HAL
open science

Sélection adaptative d'opérateurs pour la recherche locale basée sur un compromis exploration-exploitation

Nadarajen Veerapen, Jorge Maturana, Frédéric Saubion

► To cite this version:

Nadarajen Veerapen, Jorge Maturana, Frédéric Saubion. Sélection adaptative d'opérateurs pour la recherche locale basée sur un compromis exploration-exploitation. JFPC - Huitièmes Journées Francophones de Programmation par Contraintes - 2012, May 2012, Toulouse, France. pp.318-327. hal-00812844

HAL Id: hal-00812844

<https://inria.hal.science/hal-00812844>

Submitted on 12 Apr 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Sélection adaptative d'opérateurs pour la recherche locale basée sur un compromis exploration-exploitation

Nadarajen Veerapen¹ Jorge Maturana² Frédéric Saubion¹

¹ LUNAM Université, Université d'Angers, LERIA, Angers, France

² Instituto de Informática, Universidad Austral de Chile, Valdivia, Chili
prenom.nom@univ-angers.fr prenom.nom@inf.uach.cl

Résumé

Cet article traite de la sélection adaptative d'opérateurs dans le contexte de la recherche locale (LS). La similarité entre la solution candidate et les solutions de la trajectoire de recherche est considérée. De paire avec la qualité de la solution, ces deux mesures nous permettent d'évaluer la performance de chaque opérateur. Une nouvelle mesure d'utilité pour opérateurs de LS, basée sur les distances relatives entre les opérateurs, est proposée. En utilisant des méthodes basiques de sélection, nous la comparons à une mesure proposée récemment basée sur le concept de dominance Pareto. Une version adaptative de l'algorithme est également examinée (variation du biais exploration-exploitation). Les méthodes proposées sont testées sur le problème d'affectation quadratique et le problème du voyageur de commerce asymétrique.

1 Introduction

Dans cet article¹ nous nous intéressons à la recherche locale (LS) pour l'optimisation combinatoire. D'une façon générale, un algorithme de recherche locale se déplace d'une configuration à une autre, en construisant un chemin de recherche. Ces mouvements peuvent être exécutés en utilisant des opérateurs qui sélectionnent la prochaine configuration à atteindre dans le voisinage de la configuration courante. Une politique de recherche efficace requiert un compromis entre deux objectifs généralement divergents : l'exploitation et l'exploration. L'exploitation implique la convergence vers un optimum local tandis que l'exploration implique un échantillonnage adéquat de l'es-

pace de recherche. Ce compromis peut être atteint en contrôlant les opérations de bases (mouvements ou opérateurs) qui dirigent le chemin de recherche dans l'espace de recherche.

La conception d'algorithmes de recherche autonomes est devenue un domaine important de la résolution de problèmes [11]. Au cours des dernières décennies, un nombre croissant de techniques de résolution a été proposé dans l'optique de résoudre des problèmes de plus en plus complexes. Toutefois elles sont souvent difficiles à adapter et à paramétrer pour un problème donné. En fait, les outils efficaces de résolution ne sont plus à la portée des utilisateurs. La conception de solveurs toujours plus efficaces produit fréquemment des systèmes fortement complexes, qui requièrent une somme non-négligeable de connaissance d'expert, par exemple pour en sélectionner intelligemment les paramètres. Cette tendance mène donc vers un besoin croissant pour des algorithmes de configuration automatique de paramètres tels que [12, 7, 13, 3]. La configuration automatique est exécutée hors-ligne en faisant tourner l'algorithme sur des instances de test. Il est également possible de contrôler les paramètres d'un algorithme en ligne, par exemple dans le contexte de la recherche réactive [1], en intégrant des techniques d'apprentissage aux heuristiques de recherche locale. La conception de stratégies de contrôle de haut niveau est une tendance relativement récente ayant pour but de rendre les techniques d'optimisation plus abordables [5]. Une classification de ces approches est présentée dans [10].

Nous nous concentrons ici sur la sélection en ligne du prochain mouvement à appliquer dans un processus de recherche locale afin de s'adapter à la réalité du

1. Cet article est une traduction de l'article *An Exploration-Exploitation Compromise-Based Adaptive Operator Selection for Local Search*, GECCO'12 [26].

sous-ensemble de l'espace de recherche en cours d'exploration. Adaptive Pursuit [23] est un exemple d'une telle stratégie affectant une probabilité de sélection à chaque opérateurs pour les algorithmes génétiques. Sa caractéristique principale est l'utilisation du concept du gagnant remporte tout (winner-takes-all) qui se traduit par la forte augmentation de la probabilité de sélection du meilleur opérateur tout en diminuant la probabilité de sélection des autres opérateurs. De nouvelles techniques de sélection adaptative d'opérateurs ont été proposées pour les algorithmes évolutionnaires, par exemple dans [8].

Dans cet article nous présentons un mécanisme de sélection d'opérateur pour la recherche locale. Un opérateur est une combinaison d'un voisinage et d'une fonction de sélection. À chaque itération, un opérateur est sélectionné à partir d'un ensemble d'opérateurs. Les critères de sélection sont basés sur la performance antérieure des opérateurs ainsi que sur un compromis entre exploration et exploitation que le processus entend encourager. Nous présentons une méthode, brièvement étudiée dans [25], permettant d'évaluer la performance d'un opérateur basée sur sa performance relative. Le compromis souhaité est tout d'abord fixé. Nous examinons ensuite une stratégie adaptative basée sur la fréquence d'apparition des valeurs des solutions récemment obtenues.

Nos fonctions de contrôle s'adressent à des problèmes de permutation. Ceci nous permet d'utiliser des combinaisons basiques de voisinages et de fonctions de sélections simples tout en obtenant un ensemble d'opérateurs suffisamment varié.

Dans [27], les opérateurs sont sélectionnés sur le principe de la Pareto dominance. La probabilité de sélection d'un opérateur est déterminée en fonction du nombre d'opérateurs que ce même opérateur domine. Une limite de cette méthode est qu'il n'y a pas de compromis explicite entre exploration et exploitation, ce qui favorise les opérateurs au milieu du front Pareto. Cet article a pour but d'étudier une stratégie alternative basée sur un compromis explicite et se base sur les résultats initiaux présentés dans [25].

Cet article est composé de sept sections. Après cette introduction, la section 2 présente les définitions et les concepts importants : voisinage, fonction de sélection et opérateur. La section 3 décrit l'intégration de la sélection d'opérateurs dans un algorithme LS ainsi que les métriques utilisées. La section 4 explique la sélection d'opérateurs. Elle comprend également un survol des travaux connexes. Les expériences pour un compromis statique sont présentées en section 5. La section 6 quant à elle est dédiée aux expériences avec un compromis adaptatif. Enfin la dernière section présente nos conclusions.

2 Concepts généraux

Étant donné un espace de recherche \mathcal{S} et une fonction objectif $eval$ sur \mathcal{S} , une recherche locale applique des opérateurs afin d'atteindre une solution optimale. Un opérateur, étant donné une configuration de \mathcal{S} , produit une nouvelle configuration. Dans sa forme la plus simple, il est composé d'un voisinage et d'une fonction de sélection. Ce voisinage est l'ensemble des configurations atteignables à partir d'une configuration donnée. La fonction de sélection choisit une configuration à partir de cet ensemble en fonction de certains critères. Le squelette principal d'une métaheuristique de recherche locale est l'application d'un opérateur au sein d'une simple boucle. L'opérateur explore un voisinage et renvoie une solution. Selon le type de métaheuristique utilisée, une étape supplémentaire telle qu'une perturbation est introduite dans la boucle.

Les concepts de voisinage, de fonction de sélection et d'opérateur sont formalisés ci-après.

Définition. Soit \mathcal{S} l'espace de recherche. Une *relation de voisinage* ou *voisinage* est une relation binaire ir-réflexive $\mathcal{N} \subseteq \mathcal{S}^2$ sur l'espace de recherche. Dans la majorité des cas, cette relation est symétrique.

La composition et l'union de voisinages sont notées $\mathcal{N} \circ \mathcal{N}'$ et $\mathcal{N} \cup \mathcal{N}'$ respectivement. Un voisinage composé avec lui-même est noté \mathcal{N}^2 et $\mathcal{N}^{n+1} = \mathcal{N} \circ \mathcal{N}^n$.

Considérons le paysage de recherche. La relation d'ordre $<$ sur \mathcal{S} correspond à l'ordre induit par la fonction objectif du problème. Seuls les problèmes de minimisation sont considérés sans perte de généralité.

Définition. Un *sélecteur* est une fonction qui effectue une sélection sur le voisinage, éventuellement guidé par l'ordre $<$ et défini par $\sigma : \mathcal{S} \times 2^{\mathcal{S}^2} \mapsto \mathcal{S}$ (ici la sélection ne retourne qu'un voisin), tel que $(s, \sigma(s, \mathcal{N})) \in \mathcal{N}^=$ (la clôture réflexive de \mathcal{S} afin d'inclure l'identité).

Définition. Un *opérateur* est alors défini par une paire (\mathcal{N}, σ) .

3 Contrôle d'opérateurs pour la LS

Le contrôle d'opérateurs se réfère à la sélection d'un opérateur approprié à partir d'un ensemble d'opérateurs disponibles à chaque itération d'un algorithme de LS. Le comportement antérieur de chaque opérateur est enregistré et analysé afin de déterminer son utilité par rapport à la gestion du compromis entre exploitation et exploration. Le schéma 1 dépeint l'interaction entre le contrôleur et la LS. Après chaque application d'un opérateur, le contrôleur reçoit la qualité de la configuration courante ainsi que sa distance

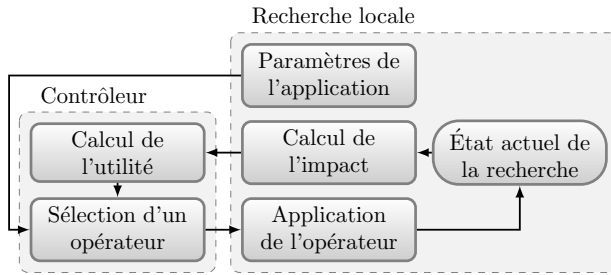


FIGURE 1 – Contrôleur et algorithme LS

par rapport au chemin de recherche. Ces métriques représentent l'impact de l'opérateur et seront décrites plus loin dans cette section. L'impact est utilisé pour calculer et mettre à jour l'utilité de chaque opérateur. Le contrôleur sélectionne ensuite le prochain opérateur à partir de ce critère. L'utilité et le processus de sélection sont décrits dans la section 4.

Quand un utilisateur souhaite résoudre un nouveau problème, il n'a qu'à définir une procédure pour lire les données de l'instance et spécifier la fonction objectif ainsi que les contraintes. L'utilisateur peut également ajouter de nouveaux opérateurs à l'ensemble d'opérateurs prédéfinis. Il est donc primordial que le contrôle d'opérateur soit suffisamment robuste pour gérer des opérateurs peu performants.

3.1 Calcul de l'impact

Comme précédemment mentionné, un aspect fondamental de la LS est le compromis entre exploitation et exploration. Des métriques adéquates sont donc nécessaires afin d'évaluer ces deux notions liées à la qualité et à la diversité respectivement. Nous utilisons ici des métriques de [27].

3.1.1 Qualité

La qualité est directement mesurée à partir de la fonction objectif. Le changement relatif de la qualité lorsqu'un opérateur op est appliqué à une solution s est donné par

$$q = \frac{eval(s) - eval(op(s))}{eval(s) + 1} \quad (1)$$

3.1.2 Distance

Mesurer la diversité est relativement simple pour les algorithmes à populations mais moins clair pour les algorithmes qui ne sont basés que sur une solution. Ici nous considérons une fenêtre glissante contenant les solutions antérieures du chemin de recherche et nous

mesurons la différence entre ces solutions et la solution candidates $c = op(s)$. Cette différence est calculée au niveau du couple variable-valeur : moins fréquentes sont les occurrences des couples variable-valeur de la solution candidate dans le chemin, plus grande est la distance est entre eux. L'équation suivante formalise cette notion. Soit $P_{i,j}$ le chemin de l'itération i jusqu'à j , $i \leq j$. Alors

$$d^P(c, P_{i,j}) = \frac{1}{n} \times \sum_{k_c=1}^n \left(1 - \frac{occ(P_{i,j}, (k_c, \pi_{k_c}))}{|P_{i,j}|} \right) \quad (2)$$

où $occ(P_{i,j}, (a, b))$ correspond au nombre de fois où le couple variable-valeur (a, b) apparaît dans $P_{i,j}$. Par exemple dans $(1, 3, 2)$, les couples sont $(1, 1)$, $(2, 3)$ et $(3, 2)$. Pour des problèmes qui modélisent des cycles, (a, b) représente deux valeurs consécutives. Par exemple dans $(1, 3, 2)$ les couples sont $(1, 3)$, $(3, 2)$ et $(2, 1)$.

Maintenant que nous avons défini comment mesurer la qualité et la distance par rapport aux solutions, la section suivante traite du calcul de l'utilité de chaque opérateur en fonction de ces mesures.

4 Sélection d'opérateurs

Dans cet article, un opérateur est sélectionné proportionnellement à valeur d'utilité représentant sa performance antérieure.

Une fenêtre glissante de taille fixe, enregistrant les valeurs de qualité et de distance sur les m dernières applications, est associée à chaque opérateur. Ces fenêtres sont initialisées en appliquant une fois chaque opérateur au début de la recherche. L'utilité d'un opérateur est calculée en se basant sur la moyenne des valeurs de qualité et de distance. Dans la suite, lorsque la qualité ou la distance d'un opérateur sera mentionnée, nous nous référerons implicitement à la valeur moyenne sur la fenêtre glissante. Nous définissons les différentes mesures d'utilité dans la sous-section suivante.

4.1 Utilité d'un opérateur

Dans [27] l'utilité de chaque opérateur est calculée en fonction du nombre d'opérateurs qu'il domine.

Définition. Soient deux vecteurs u et v de même cardinalité p et considérant un problème de maximisation, u domine v si $u_k \geq v_k, \forall k \in \{1, \dots, p\}$ avec au moins une inégalité stricte. Ceci est souvent appelé la Pareto dominance et notée $u \succ v$.

L'utilité basée sur la Pareto dominance U_P de l'opérateur o dans l'ensemble O de n opérateurs est alors

définie par

$$U_P(o) = |\{o' | o' \in O, o \succ o'\}| + \epsilon \quad (3)$$

où ϵ est une valeur positive afin d'obtenir une utilité non nulle.

Cette utilité ne permet pas de spécifier un compromis entre exploitation et exploration explicitement. Nous introduisons une pondération, α , afin d'influencer ce compromis.

Soient deux opérateurs o_1 and o_2 définis par les couples qualité-distance (q_1, d_1) et (q_2, d_2) , nous définissons le déplacement rectilinéaire pondéré de o_1 vers o_2 tel que

$$d_\alpha(o_1, o_2) = \alpha(d_1 - d_2) + (1 - \alpha)(q_1 - q_2) \quad (4)$$

À partir de cette métrique, nous définissons la nouvelle utilité : somme des déplacements positifs vers les autres opérateurs.

$$U_\alpha^{\Sigma+}(o) = \sum_{i=1}^n \max(0, d_\alpha(o, o_i)) \quad (5)$$

Comme point de comparaison, nous définissons également la simple somme pondérée de la qualité et de la distance.

$$U_\alpha(o) = \alpha d + (1 - \alpha)q \quad (6)$$

La figure 2 tente d'illustrer l'intérêt d'utiliser la somme des déplacements comme valeur d'utilité d'un opérateur. Primo, les déplacements, et leur somme, sont utiles puisqu'ils décrivent quantitativement (la magnitude) et, dans une moindre mesure, qualitativement (le signe ou la direction) la relation entre opérateurs. Secundo, la pondération introduit naturellement un biais quantifiable vers l'exploration ou l'exploitation.

4.2 Travaux connexes

Parmi un grand nombre de travaux [11], la conception d'algorithmes de recherche locale autonome a été traitée dans le contexte de la recherche réactive [1] dans des travaux fondateurs tels que le tabou réactif [2], le recuit simulé adaptatif [14] ou la recherche locale adaptative [6]. En général seul un critère, la qualité de la solution, est considéré. Ces travaux se concentrent sur des paramètres correspondants aux heuristiques spécifiques de ces méthodes, par exemple la taille de la liste tabou.

Dans cet article, nous souhaitons manipuler simultanément le voisinage et la fonction de sélection en tant qu'entité unique : l'opérateur. De plus, nous souhaitons également utiliser de multiples critères génériques

(la qualité et la diversité) afin d'orienter la politique de recherche.

En ce qui concerne le contrôle de paramètres, les techniques les plus avancées ont été initialement développées dans le contexte de l'optimisation évolutionnaire [15]. Comme mentionné précédemment, Adaptive Pursuit [23] affecte une probabilité à chaque opérateur pour les algorithmes génétiques en utilisant une stratégie du gagnant remporte tout (winner-takes-all).

Dans sa thèse de doctorat, Fialho [8] décrit un nombre de stratégies de sélection (sélection adaptative d'opérateurs) plus récentes pour les algorithmes génétiques, dont le bandit manchot multibras dynamique qui sélectionne un opérateur qui maximise la somme de deux valeurs. La première représente la performance de l'opérateur et la seconde assure que l'opérateur sera sélectionné un nombre infini de fois. Il présente également plusieurs mesures pour déterminer l'utilité des opérateurs tel que l'AUC (l'aire sous la courbe) pour un critère (la qualité de la population). Dans [19, 18], des techniques de sélection d'opérateurs sont proposées afin de traiter deux critères simultanément dans l'évaluation des opérateurs : la qualité et la diversité de la population. Dans le contexte de la recherche locale, ceci a également été tenté dans [27], offrant des résultats préliminaires.

Plus récemment, un solveur fortement paramétrable pour SAT a été présenté dans [24] où la probabilité d'utiliser un voisinage ou une fonction de sélection est déterminée par un réglage hors-ligne intensif. Si nous considérons la LS associée au paradigme de programmation par contrainte, [17] présente un concept apparenté à la sélection adaptative d'opérateurs pour la recherche à grand voisinage pour le problème d'ordonnement de véhicules. Des résultats initiaux pour une version améliorée basée sur l'apprentissage par renforcement sont présentés dans [16]. Par rapport à ces précédentes approches en LS, nous avons comme objectif de proposer des méthodes plus sophistiquées pour l'évaluation de l'utilité des opérateurs ainsi que des techniques de sélection plus adaptatives. Toutefois, vue la spécificité à un problème donné des méthodes précédemment citées, la comparaison avec notre méthode est problématique. Nous entendons traiter cet aspect dans un prochain travail, en élargissant la gamme de problèmes traités.

5 Expériences pour l'utilité pondérée

Le codage des configurations correspond à un problème de permutation. Si N est la taille du problème (le nombre de variables), une solution est représentée par une permutation de N entiers $\{1, 2, \dots, N\}$, c.-à-d. que chaque variable a le même domaine et deux va-

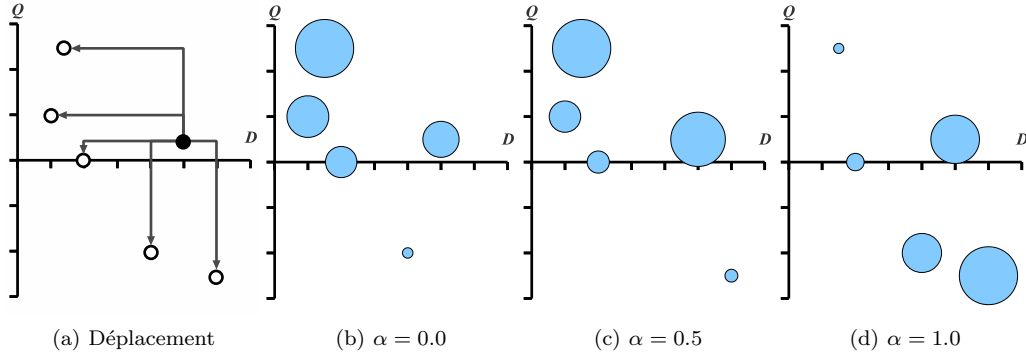


FIGURE 2 – (a) Une population hypothétique de six opérateurs montrant les déplacements non pondérés relativement à un opérateur; (b)–(d) α détermine l'importance (taille de la bulle) de chaque opérateur pour $U_{\alpha}^{\Sigma+}$.

riables ne peuvent avoir la même valeur.

Soit Π l'espace de recherche, c.-à-d. l'ensemble de toutes les permutations de l'ensemble $\{1, 2, \dots, N\}$. Si $\pi \in \Pi_N$ et $1 \leq i \leq N$, alors π_i correspond à l'élément i dans π .

Les différentes mesures d'utilité décrites dans la section précédente sont tout d'abord testées sur un problème classique de permutation : le problème d'affectation quadratique (QAP) dans lequel il faut trouver le coût minimum lorsque que N unités de productions sont placés dans N lieux, le coût étant la somme de tous les produits distance-flot possibles. La fonction objectif est donnée par

$$\min_{\pi \in \Pi_N} \sum_{i=1}^N \sum_{j=1}^N a_{\pi_i, \pi_j} b_{i,j} \quad (7)$$

Afin d'évaluer la robustesse des méthodes proposées, des tests sont ensuite basés sur le problème du voyageur de commerce asymétrique (ATSP) qui revient à trouver le plus court circuit entre N villes (le plus petit cycle hamiltonien) étant donné la distance $d(c_i, c_j)$ pour chaque paire de villes (c_i, c_j) . La fonction objectif est donnée par

$$\min_{\pi \in \Pi_N} \sum_{i=1}^{N-1} d(c_{\pi_i}, c_{\pi_{i+1}}) + d(c_{\pi_N}, c_{\pi_1}) \quad (8)$$

Les expériences sur l'ATSP sont décrites à la section 6.2.

5.1 Cadre expérimental

Chaque opérateur est une combinaison d'un voisinage et d'une fonction de sélection. Nous utilisons un voisinage de base, \mathcal{N}_E (échange). Il est défini tel que $(s, s') \in \mathcal{N}_E$ si et seulement si $s =$

$(\pi_1, \dots, \pi_{i-1}, \pi_i, \pi_{i+1}, \dots, \pi_{j-1}, \pi_j, \pi_{j+1}, \dots, \pi_n)$ et $s' = (\pi_1, \dots, \pi_{i-1}, \pi_j, \pi_{i+1}, \dots, \pi_{j-1}, \pi_i, \pi_{j+1}, \dots, \pi_n)$ pour i et j quelconques. De plus grands voisinages sont créés par composition de ce voisinage avec lui-même.

Le tableau 1 présente les différents sélecteurs et opérateurs utilisés. Parmi les 12 opérateurs, 10 d'entre eux (O1–O10) sont des opérateurs modérés, c.-à-d. qu'ils ne modifient qu'un petit nombre de variables. Les deux derniers (O11 et O12) sont des opérateurs perturbateurs modifiant 25% et 50% des variables respectivement. Les opérateurs O1–O5 sont orientés exploitation tandis que les O6–O12 sont orientés exploration. Le mécanisme de sélection d'opérateurs n'a pas connaissance de ces informations.

Les opérateurs perturbateurs sont inclus afin de pénaliser la sélection uniforme et d'augmenter la difficulté de la tâche des mécanismes de sélection. En effet, des travaux précédents [27] ont montrés que si la population d'opérateurs est raisonnablement bien sélectionnée, la sélection uniforme parvient à produire des résultats de qualité raisonnable. De plus, ces opérateurs perturbateurs peuvent être considérés comme des mécanismes de redémarrage dynamiques puisque les opérateurs ne sont pas appliqués dans un ordre prédéterminé.

Tel que mentionné précédemment, les opérateurs sont simplement sélectionnés proportionnellement à leur utilité (noté PR plus loin). Nous comparons cette méthode de sélection à une autre méthode légèrement plus complexe : Adaptive Pursuit [23].

Adaptive Pursuit (AP) emploie une stratégie de gagnant remporte tout. Une probabilité minimum p_{min} est utilisée ainsi qu'un paramètre α' qui contrôle l'importance des valeurs précédemment obtenues. Plus α' est grand, plus la nouvelle valeur a de l'importance. Le

TABLE 1 – Définitions des sélecteurs et des opérateurs

| Sélecteur | Nom | Définition |
|---|--------------------------|--|
| σ_R | Aléatoire | $\sigma_R(s, \mathcal{N})$ est n'importe quel s' tel que $(s, s') \in \mathcal{N}$ |
| σ_I | Amélioration | $\sigma_I(s, \mathcal{N})$ est n'importe quel s' tel que $(s, s') \in \mathcal{N}$ and $s' < s$ |
| σ_{BI} | Meilleure amélioration | $\sigma_{BI}(s, \mathcal{N})$ est un élément minimal s' selon l'ordre $<$, tel que $(s, s') \in \mathcal{N}$ |
| σ_{BIk} | Meilleure amélioration k | $\sigma_{BIk}(s, \mathcal{N})$ est un élément uniformément sélectionné $s' \in K$, K étant l'ensemble des k -meilleurs éléments selon l'ordre $<$, tel que $(s, s') \in \mathcal{N}$ |
| σ_{Tk} | Tournois k | $\sigma_{Tk}(s, \mathcal{N})$ est un élément s' tel que K est le sous-ensemble de k éléments en relation avec s dans \mathcal{N} et s' est le meilleur de ces k éléments |
| Opérateur | Nom | Définition |
| $(\sigma_I, \mathcal{N}_E)$ | O1 | première amélioration pour l'échange entre deux variables. |
| $(\sigma_{BI}, \mathcal{N}_E)$ | O2 | meilleure amélioration pour l'échange entre deux variables. |
| $(\sigma_{BI5}, \mathcal{N}_E)$ | O3 | choix aléatoire parmi les cinq meilleurs échanges entre deux variables. |
| $(\sigma_{BI}, \mathcal{N}_E)^2$ | O4 | deux meilleurs échanges consécutifs ; les variables échangées à l'étape une sont interdites à l'étape deux. |
| $(\sigma_{BI}, \mathcal{N}_E)^3$ | O5 | trois meilleurs échanges consécutifs ; les variables échangées aux étapes précédentes sont interdites à l'étape suivante. |
| $(\sigma_{T3}, \mathcal{N}_E^2)$ | O6 | meilleur échange entre trois variables choisies aléatoirement. |
| $(\sigma_{T4}, \mathcal{N}_E^3)$ | O7 | meilleur échange entre quatre variables choisies aléatoirement. |
| $(\sigma_{T5}, \mathcal{N}_E^4)$ | O8 | meilleur échange entre cinq variables choisies aléatoirement. |
| $(\sigma_{T6}, \mathcal{N}_E^5)$ | O9 | meilleur échange entre six variables choisies aléatoirement. |
| $(\sigma_R, \mathcal{N}_E)^3$ | O10 | trois échanges consécutifs entre deux variables. |
| $(\sigma_R, \mathcal{N}_E)^{\lfloor n/8 \rfloor}$ | O11 | $\lfloor n/8 \rfloor$ échanges aléatoires consécutifs entre deux variables. |
| $(\sigma_R, \mathcal{N}_E)^{\lfloor n/4 \rfloor}$ | O12 | $\lfloor n/4 \rfloor$ échanges aléatoires consécutifs entre deux variables. |

paramètre β définit le niveau de « gloutonnerie ». Parmi k opérateurs la probabilité de sélectionner l'opérateur i à l'itération t est donnée par

$$p_{i,t} = \begin{cases} p_{i,t-1} + \beta(p_{max} - p_{i,t-1}) & \text{if } i = i_{t-1}^* \\ p_{i,t-1} + \beta(p_{min} - p_{i,t-1}) & \text{otherwise} \end{cases} \quad (9)$$

où

$$i_t^* = \operatorname{argmax}_{i=1\dots k} \{Q_{i,t}\} \text{ et } p_{max} = 1 - (k-1)p_{min}$$

et

$$Q_{i,t} = (1 - \alpha')Q_{i,t-1} + \alpha'U_{i,t-1}$$

L'utilité basée sur la Pareto dominance est utilisée pour AP avec $\epsilon = 0$ puisque p_{min} joue un rôle équivalent.

Le paramètre pour U_P est $\epsilon = 0.1$ (les valeurs 0.01, 0.05, 0.2 et 1 ont également été testées mais 0.1 a obtenu les meilleurs résultats sur une majorité d'instances). Pour $U_\alpha^{\Sigma+}$ et U_α , le paramètre α a été testé avec les valeurs 0.2, 0.5 et 0.8.

La méthode de réglage F-Race [3] a été utilisée pour déterminer la valeur des paramètres de AP ($\alpha' = 0.9$, $\beta = 0.1$, $p_{min} = 0.01$). Les combinaisons de paramètres testées ont été générées à partir de $\{0.1, 0.3, 0.6, 0.9\}$ pour α' et β et $\{0, 0.001, 0.01, 0.1, 0.1\}$ pour p_{min} .

Pour chaque instance, chaque algorithme est lancé 30 fois et un maximum de 40 000 itérations lui sont accordées par exécution. Pour le QAP, les mêmes 30 solutions aléatoires sont utilisées au début de chaque algorithme. Pour l'ATSP la solution initiale est calculée de façon gloutonne. La longueur de toutes les fenêtres glissantes est fixée à 100 (pour le chemin et les mesures de qualité et de distance pour chaque opérateur).

5.2 Analyse des résultats

Le tableau 2 présente les résultats concernant toutes les expériences sur des instances de QAP issues de QAPLIB [4] dont parle cet article. Les valeurs pour chaque algorithme représentent le pourcentage moyen de différence par rapport aux meilleures valeurs connues. Dans cette section nous nous intéressons à toutes les colonnes sauf l'avant-dernière. Les résultats pour U_α (disponibles dans [25]) ne sont pas présentés dans le tableau puisqu'ils n'offrent pas de meilleurs résultats que U_P pour les valeurs α testées. La dernière colonne donne, à titre de comparaison, les résultats obtenus pour le tabou robuste (RTS) [22], un algorithme LS dédié au QAP. RTS est un algorithme relative-

ment ancien mais a été choisi car il demeure à ce jour l’algorithme ayant trouvé le plus grand nombre de meilleures solutions connues pour les instances de QAPLIB n’ayant pas été résolues de façon optimale. Les chiffres en gras indiquent les meilleures valeurs et en italique les résultats à 0.05 % de la meilleure valeur (les résultats de RTS ne sont pas considérés puisqu’ils sont toujours meilleurs ou équivalents).

Considérons tout d’abord la qualité générale moyenne des solutions obtenues avec toutes les méthodes. Toutes, même la sélection uniforme, sont généralement à un ou deux pour cents de l’optimale ou de la meilleure valeur connue. Ceci démontre que le nombre alloué d’itérations n’est pas un facteur limitant.

Le tableau 2 nous permet d’observer que U_P (un paramètre) avec une sélection dont la probabilité est directement calculée à partir de l’utilité (PR dans le tableau) se comporte très bien par rapport à AP qui a trois paramètres. Pour AP, le réglage des paramètres ne lui permet que de se comporter à un niveau équivalent ou pire que PR. Le concept du gagnant remporte tout, utile pour les algorithmes évolutionnaires, ne semble pas apporter de bénéfice dans les conditions de test utilisées ici. Au lieu de donner l’avantage au meilleur opérateur, il semble qu’il soit plus approprié d’avoir une probabilité plus élevée de choisir parmi quelques opérateurs qui se comportent raisonnablement bien.

Si nous comparons U_P (PR) et $U_\alpha^{\Sigma+}$, il en ressort que la seconde est généralement équivalente à la première pour un petit α et vraiment meilleur sur les instances *taixxa*. À partir de cette observation, et parce que différentes classes d’instances semblent nécessiter différents α pour obtenir les meilleurs résultats, une méthode pour faire varier α et améliorer les résultats est décrite dans la section suivante.

U_α est la seule utilité à ne pas être calculé à partir d’une relation entre opérateurs. Ainsi que mentionné précédemment, ses résultats sont pire que U_P . Il semble donc que l’utilisation d’une simple somme pondérée est trop simpliste et que considérer la relation entre opérateurs peut être utile lorsqu’elle est utilisée de façon appropriée.

Nous avons inclus RTS comme point de comparaison afin d’illustrer les possibilités d’amélioration pour les méthodes génériques. Néanmoins, nous ne nous attendions pas à dépasser un algorithme dédié basé sur des prohibitions en utilisant une méthode générique utilisant des opérateurs génériques pour des problèmes de permutation, dont certains étant très perturbateurs.

La section suivante se penche sur une méthode pour faire varier α au fil de la recherche afin d’obtenir de meilleurs résultats.



FIGURE 3 – Les modules Stratégie, Contrôleur et LS

6 Valeurs adaptatives de paramètres pour le contrôle d’opérateurs

Dans la section précédente nous avons utilisé un compromis statique. Nous examinons maintenant comment ce compromis peut s’adapter en ligne à l’état du processus de recherche.

6.1 Stratégie de réglage des paramètres

Afin de faire varier α , nous introduisons le module Stratégie qui se fixe au contrôleur comme illustré sur le schéma 3. Il communique avec ce dernier et son rôle est de faire varier les paramètres en direct.

Dans cet article nous considérons une stratégie de diversité « correcte » (CD), un concept développé dans [20] pour les algorithmes évolutionnaires. La stratégie CD utilise la qualité des solutions de la population comme moyen d’évaluation de la diversité de la population. Si le nombre de solutions ayant la même qualité est au-dessus d’un certain seuil T_{max} alors on suppose que la population est trop homogène et la diversité commandée est incrémentée d’un pas s_{inc} . De façon symétrique, si le nombre de solutions ayant la même qualité est au-dessous d’un autre seuil T_{min} la diversité commandée est décrétementée d’un pas s_{dec} , la supposition étant que la population est incapable de rester proche d’optima locaux de bonne qualité, ce qui démontre une faible exploitation. Dans notre framework, CD utilise les solutions présentes dans la fenêtre glissante, la même fenêtre qui est utilisée pour le calcul de la distance.

6.2 Cadre expérimental

Nous avons utilisé F-Race afin d’effectuer le réglage des paramètres. Les valeurs testées proviennent des domaines suivants : $T_{max} = \{0.3, 0.4, 0.5, 0.6\}$, $T_{min} = \{0.05, 0.1, 0.15, 0.20, 0.25\}$, and $\{0.0001, 0.001, 0.01, 0.1\}$ for s_{inc} et s_{dec} . Ceci donne 320 combinaisons. Le gagnant est ($T_{max} = 0.3$, $T_{min} = 0.25$, $s_{inc} = 0.0001$, $s_{dec} = 0.1$).

La combinaison gagnante étant à une extrémité de chaque domaine, on peut être amené à penser qu’une meilleure combinaison peut être obtenue en élargissant les domaines. Nous avons donc lancé un deuxième réglage avec de nouveaux domaines et obtenu un nouveau vainqueur ($T_{max} = 0.35$, $T_{min} = 0.15$, $s_{inc} =$

0.0001, $s_{dec} = 0.01$). Ce dernier est relativement différent du précédent mais n'a pas bénéficié des nouvelles valeurs aux extrémités de chaque domaine.

Lorsque les deux gagnants sont testés côte à côte, leurs distributions de résultats sont statistiquement équivalentes. Ceci nous amène à penser que la stratégie n'est pas hypersensible à sa configuration de paramètres et est donc relativement robuste.

En plus du QAP, l'ATSP est également utilisé dans cette section (instances provenant de TSPLIB [21]). Le scénario de test pour l'ATSP réutilise les paramètres trouvés pour le QAP (test de robustesse). De nouvelles combinaisons de paramètres sont ensuite réglées pour l'ATSP et comparées.

De plus, les mêmes opérateurs que pour le QAP sont conservés bien qu'ils ne soient pas réellement adaptés à la résolution du l'ATSP. Ceci teste l'aptitude de la méthode à surmonter des opérateurs mal choisis. Un opérateur supplémentaire spécifique à l'ATSP est ajouté afin d'être en mesure de produire des solutions améliorantes. Le 3-opt [9] ($\sigma_{BI}, \mathcal{N}_{EE}^2$) sélectionne la meilleure solution obtenue en supprimant trois côtés et en reconstruisant trois nouveaux de sorte qu'aucun sous-chemin ne soit inversé. \mathcal{N}_{EE} est défini par $(s, s') \in \mathcal{N}_{EE}$ ssi $s = (\pi_1, \dots, \pi_{i-1}, \pi_i, \pi_{i+1}, \dots, \pi_{j-1}, \pi_j, \pi_{j+1}, \dots, \pi_n)$ et $s' = (\pi_1, \dots, \pi_i, \pi_j, \pi_{j-1}, \dots, \pi_{i+1}, \pi_{j+1}, \dots, \pi_n)$ pour $i + 1 < j$.

Dans la suite nous examinons la performance de la première combinaison gagnantes par rapport aux résultats précédemment obtenus.

6.3 Analyse des résultats

Les nouveaux résultats pour le QAP sont présentés dans la tableau 2, à la colonne $U_\alpha^{\Sigma+}CD$. Il semble claire que CD est meilleure que les autres méthodes, ou du moins à 0.05 % du meilleur résultats, sur la majorité des instances. Cette supériorité est confirmée par un test de rangs signés de Wilcoxon avec un seuil de confiance de 95 %. Si nous comparons $U_\alpha^{\Sigma+}CD$ aux meilleures valeurs sur les différents α pour $U_\alpha^{\Sigma+}$, les deux distributions sont statistiquement équivalentes. Ceci nous permet de conclure que la stratégie CD est suffisamment performante pour produire des résultats équivalents aux meilleurs résultats de $U_\alpha^{\Sigma+}$ avec un α préréglé.

Les résultats pour l'ATSP sont donnés dans le tableau 3. Les colonnes 4 et 6 utilisent les paramètres choisis pour le QAP. Les colonnes 3, 5 et 7 (astérisques dans les entêtes ds colonnes) utilisent des paramètres réglés pour l'ATSP : $\epsilon = 0.01$ pour U_P avec PR, ($\alpha = 0.3, \beta = 0.1, p_{min} = 0.001$) pour AP, et ($T_{max} = 0.6, T_{min} = 0.25, s_{inc} = 0.0001, s_{dec} = 0.01$) pour CD.

Si l'on considère les résultats pour le paramétrage spécifique au QAP, les différentes méthodes parviennent à gérer la population d'opérateurs très défavorable. Pour preuve, les résultats pour la sélection uniforme sont très mauvais et rappelons-nous qu'il n'y a qu'un seul opérateur qui soit un bon opérateur pour l'ATSP. D'autre part, AP est peu performante sur les plus grandes instances. Bien entendu, de meilleurs résultats bruts auraient été obtenus si la population d'opérateurs avait été pensée pour l'ATSP. Comme avec le QAP, la stratégie CD produits de bons résultats mais ici elle ne parvient à dépasser les autres méthodes que sur les plus petites instances.

Considérons maintenant les résultats pour le paramétrage spécifique à l'ATSP. Faute de place les résultats pour U_P avec PR ne sont pas inclus mais sont disponibles dans [26]. Par rapport à PR, PR* produit une amélioration sur les plus grandes instances contrebalancée par des résultats de qualité inférieure sur les autres instances. De fait, PR* n'est pas une amélioration en général mais sert plutôt à souligner l'importance du paramètre ϵ . Une valeur plus petite signifie une faible probabilité de sélection des opérateurs très perturbateurs et donc une meilleure performance sur les plus grandes instances. Pour U_P avec AP, AP* donne une amélioration significative. Cette méthode de sélection est très sensible à la configuration de ses paramètres et à la population d'opérateurs. En effet, ici il y a un seul opérateur qui puisse bénéficier de la stratégie du gagnant remporte tout. Néanmoins, les résultats sont équivalents à ceux de la sélection PR* qui est bien plus simple.

Les nouveaux paramètres pour CD ne semblent pas apporter de réel changement positif ou négatif. La faible performance de CD sur les plus grandes instances provient du fait qu'à partir du moment où un fort biais vers l'exploration apparaît, ce biais ne diminue pas suffisamment rapidement, ce qui implique trop d'applications des opérateurs perturbateurs.

La stratégie CD soit basée sur plusieurs paramètres, toutefois les résultats pour CD et CD* montrent que l'algorithme n'est pas hypersensible à leurs valeurs. CD est robuste. Bien que les paramètres demandent un réglage, les résultats tendent à montrer que ce réglage n'est pas forcément à refaire lorsqu'un nouveau problème se présente.

7 Conclusion

Dans cet article nous avons présenté différentes méthodes pour la sélection d'opérateurs en recherche locale. Les opérateurs étaient sélectionnés à partir de valeurs d'utilité calculées en fonction de deux critères : la qualité et la distance par rapport à la trajectoire de

TABLE 2 – QAP – le pourcentage moyen de différence par rapport aux meilleures valeurs connues.

| Inst. | Unif. | U_P | | $U_\alpha^{\Sigma+}$ | $U_\alpha^{\Sigma+}$ | RTS |
|--------|-------|-------------|-------|--|----------------------|------|
| | | PR | AP | $\alpha _{\substack{0.2 \\ 0.5 \\ 0.8}}$ | CD | |
| bur26a | 0.03 | 0.00 | 0.02 | $\begin{matrix} 0.02 \\ 0.01 \\ \mathbf{0.00} \end{matrix}$ | 0.01 | 0.00 |
| chr25a | 20.2 | 10.7 | 13.82 | $\begin{matrix} 13.8 \\ 14.9 \\ 12.1 \end{matrix}$ | 12.5 | 7.1 |
| kra30a | 2.49 | 0.79 | 1.54 | $\begin{matrix} 1.57 \\ 1.63 \\ 0.89 \end{matrix}$ | 0.61 | 0.06 |
| kra30b | 1.11 | 0.21 | 0.30 | $\begin{matrix} 0.16 \\ 0.45 \\ 0.32 \end{matrix}$ | 0.13 | 0.02 |
| nug20 | 0.12 | 0.01 | 0.01 | $\begin{matrix} \mathbf{0.00} \\ 0.03 \\ \mathbf{0.00} \end{matrix}$ | 0.01 | 0.00 |
| nug30 | 1.24 | 0.20 | 0.32 | $\begin{matrix} 0.31 \\ 0.19 \\ 0.39 \end{matrix}$ | 0.11 | 0.01 |
| sko42 | 2.28 | 0.29 | 0.38 | $\begin{matrix} 0.19 \\ 0.28 \\ 0.67 \end{matrix}$ | 0.16 | 0.03 |
| sko49 | 2.48 | 0.36 | 0.43 | $\begin{matrix} \mathbf{0.21} \\ 0.27 \\ 0.81 \end{matrix}$ | 0.24 | 0.13 |
| tai30a | 2.59 | 1.26 | 1.50 | $\begin{matrix} 1.17 \\ 1.27 \\ 1.68 \end{matrix}$ | 0.91 | 0.51 |
| tai50a | 4.20 | 2.16 | 2.13 | $\begin{matrix} \mathbf{1.58} \\ 1.59 \\ 2.83 \end{matrix}$ | 1.66 | 1.39 |
| tai30b | 0.43 | 0.13 | 0.25 | $\begin{matrix} 0.44 \\ 0.35 \\ 0.16 \end{matrix}$ | 0.15 | 0.03 |
| tai50b | 2.36 | 0.25 | 0.36 | $\begin{matrix} 0.30 \\ 0.39 \\ 0.37 \end{matrix}$ | 0.18 | 0.15 |
| wil50 | 1.24 | 0.16 | 0.15 | $\begin{matrix} 0.09 \\ 0.11 \\ 0.28 \end{matrix}$ | 0.08 | 0.05 |

recherche. Une contribution de cet article est l'introduction d'une nouvelle utilité pondérée qui autorise un compromis entre exploration et exploitation. En utilisant une pondération statique, cette utilité s'est montrée compétitive par rapport à l'utilité basée sur la Pareto dominance. Une stratégie adaptative pour faire varier la pondération a été étudiée et a offert une amélioration des résultats.

Comme perspectives de recherche, nous souhaitons étudier de nouvelles stratégies adaptative de sélection d'opérateurs. Par exemple, comme cela a été montré sur l'ATSP, la réduction linéaire du poids α n'est pas une stratégie optimale et pourrait être améliorée. Nous souhaitons également augmenter le nombre de problèmes testés et notamment passer aux problèmes booléens. Un autre objectif est la comparaison avec des méthodes de sélection où la probabilité de sélection des opérateurs est réglée hors-ligne, ce qui nécessite un temps de calcul conséquent.

Note. Ces travaux sont financés par Microsoft Research à travers le PhD Scholarship Programme ainsi que par CONICYT-ECOS (Projet C10E07).

TABLE 3 – ATSP – pourcentage moyen de différence par rapport aux meilleures valeurs connues.

| Inst. | Unif. | U_P | | U_P | $U_\alpha^{\Sigma+}$ | $U_\alpha^{\Sigma+}$ |
|---------|-------|------------|------|------------|----------------------|----------------------|
| | | PR* | AP | AP* | CD | CD* |
| ry48p | 14.0 | 1.5 | 2.4 | 1.9 | 0.6 | 0.7 |
| ft53 | 32.5 | 2.0 | 1.1 | 2.5 | 0.9 | 0.7 |
| ft70 | 10.8 | 0.6 | 1.3 | 0.7 | 0.8 | 0.7 |
| ftv44 | 23.8 | 2.3 | 2.5 | 2.6 | 0.9 | 1.0 |
| ftv47 | 27.5 | 1.5 | 1.7 | 1.1 | 0.5 | 0.5 |
| ftv55 | 22.5 | 2.8 | 3.5 | 3.2 | 1.2 | 1.3 |
| ftv64 | 32.5 | 3.5 | 4.4 | 3.1 | 1.7 | 1.6 |
| ftv70 | 26.9 | 3.9 | 5.1 | 3.5 | 2.2 | 2.0 |
| ftv90 | 30.8 | 6.0 | 10.7 | 4.9 | 4.0 | 4.4 |
| ftv100 | 32.5 | 4.8 | 14.2 | 5.0 | 4.4 | 5.3 |
| ftv120 | 38.4 | 6.8 | 21.7 | 6.7 | 8.6 | 8.5 |
| ftv140 | 43.0 | 8.3 | 29.6 | 8.4 | 11.1 | 12.6 |
| ftv160 | 42.4 | 8.3 | 37.7 | 8.2 | 15.1 | 16.6 |
| kro124p | 28.0 | 4.5 | 8.7 | 3.6 | 3.4 | 3.4 |
| rbg323 | 31.7 | 0.1 | 12.3 | 0.0 | 3.3 | 3.0 |
| rbg403 | 36.2 | 0.0 | 3.2 | 0.0 | 0.3 | 0.3 |
| rbg443 | 39.9 | 0.0 | 4.5 | 0.0 | 1.0 | 0.6 |

Références

- [1] R. Battiti, M. Brunato, and F. Mascia. *Reactive Search and Intelligent Optimization*, volume 45 of *Operations research/Computer Science Interfaces*. Springer Verlag, 2008.
- [2] R. Battiti and G. Tecchiolli. The Reactive Tabu Search. *INFORMS JOURNAL ON COMPUTING*, 6(2) :126–140, January 1994.
- [3] M. Birattari. *The Problem of Tuning Metaheuristics as seen from a machine learning perspective*. PhD thesis, Université Libre de Bruxelles, Brussels, Belgium, December 2004.
- [4] R.E. Burkard, S.E. Karisch, and F. Rendl. QAPLIB – A Quadratic Assignment Problem Library. *Journal of Global Optimization*, 10 :391–403, 1997.
- [5] E.K. Burke, G. Kendall, J. Newall, E. Hart, P. Ross, and S. Schulenburg. Hyper-Heuristics : An Emerging Direction in Modern Search Technology. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Ma-*

- agement Science, pages 457–474. Springer New York, 2003.
- [6] P. Codognot and D. Diaz. Yet Another Local Search Method for Constraint Solving. In K. Steinhöfel, editor, *Stochastic Algorithms : Foundations and Applications*, volume 2264 of *Lecture Notes in Computer Science*, pages 342–344. Springer Berlin / Heidelberg, 2001.
- [7] A. E. Eiben and S. K. Smit. Evolutionary Algorithm Parameters and Methods to Tune Them. In Hamadi et al. [11], pages 15–36.
- [8] Á. Fialho. *Adaptive Operator Selection for Optimization*. PhD thesis, Université Paris-Sud 11, Orsay, France, December 2010.
- [9] G. Gutin and A.P. Punnen, editors. *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.
- [10] Y. Hamadi, É. Monfroy, and F. Saubion. What Is Autonomous Search? In P. van Hentenryck and M. Milano, editors, *Hybrid Optimization*, volume 45 of *Springer Optimization and Its Applications*, pages 357–391. Springer New York, 2011.
- [11] Y. Hamadi, É. Monfroy, and F. Saubion, editors. *Autonomous Search*. Springer Berlin Heidelberg, 2012.
- [12] H.H. Hoos. Automated Algorithm Configuration and Parameter Tuning. In Hamadi et al. [11], pages 37–71.
- [13] F. Hutter, H.H. Hoos, K. Leyton-Brown, and T. Stützle. ParamILS : an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36(1) :267–306, September 2009.
- [14] L. Ingber. Adaptive Simulated Annealing (ASA) : Lessons learned. *Control and Cybernetics*, 25 :33–54, 1996.
- [15] F.G. Lobo, C.F. Lima, and Z. Michalewicz, editors. *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*. Springer, 2007.
- [16] J.B. Mairry, Y. Deville, and P. Van Hentenryck. Reinforced Adaptive Large Neighborhood Search. In *Eighth International Workshop on Local Search Techniques in Constraint Satisfaction (LSCS2011)*. A Satellite Workshop of CP 2011, Italy, 2011.
- [17] J.B. Mairry, P. Schaus, and Y. Deville. Generic Adaptive Heuristics for Large Neighborhood Search. In *Seventh International Workshop on Local Search Techniques in Constraint Satisfaction (LSCS2010)*. A Satellite Workshop of CP 2010, Scotland, 2010.
- [18] J. Maturana, Á. Fialho, F. Saubion, M. Schoenauer, and M. Sebag. Extreme compass and Dynamic Multi-Armed Bandits for Adaptive Operator Selection. In *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, pages 365–372, 2009.
- [19] J. Maturana, F. Lardeux, and F. Saubion. Autonomous operator management for evolutionary algorithms. *Journal of Heuristics*, 16 :881–909, 2010.
- [20] J. Maturana and F. Saubion. On the Design of Adaptive Control Strategies for Evolutionary Algorithms. In N. Monmarché, E.G. Talbi, P. Collet, M. Schoenauer, and E. Lutton, editors, *Artificial Evolution*, volume 4926 of *Lecture Notes in Computer Science*, pages 303–315. Springer Berlin / Heidelberg, 2008.
- [21] G. Reinelt. TSPLIB - a traveling salesman problem library. *INFORMS JOURNAL ON COMPUTING*, 3(4) :376–384, January 1991.
- [22] É.D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17(4-5) :443–455, 1991.
- [23] D. Thierens. An adaptive pursuit strategy for allocating operator probabilities. In *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, GECCO '05*, pages 1539–1546. ACM, 2005.
- [24] D.A.D. Tompkins, A. Balint, and H.H. Hoos. Captain Jack : New Variable Selection Heuristics in Local Search for SAT. In Karem A. Sakallah and Laurent Simon, editors, *Theory and Applications of Satisfiability Testing - SAT 2011*, volume 6695, pages 302–316. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [25] N. Veerapen, J. Maturana, and F. Saubion. A Comparison of Operator Utility Measures for Online Operator Selection in Local Search. In *Proceedings of the 6th Learning and Intelligent Optimization Conference (LION6)*, Paris, France, January 2012. À paraître.
- [26] N. Veerapen, J. Maturana, and F. Saubion. An Exploration-Exploitation Compromise-Based Adaptive Operator Selection for Local Search. In *Genetic and Evolutionary Computation Conference (GECCO'12)*, Philadelphie, USA, July 2012. À paraître.
- [27] N. Veerapen and F. Saubion. Pareto Autonomous Local Search. In C. Coello, editor, *Learning and Intelligent Optimization*, volume 6683 of *Lecture Notes in Computer Science*, pages 392–406. Springer Berlin / Heidelberg, 2011.