UQAC
UNIVERSITÉ DU QUÉBEC
À CHICOUTIMI

MÉMOIRE

PRÉSENTÉ À

L'UNIVERSITÉ DU QUÉBEC À CHICOUTIMI

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

ANNIE LEBEL

ALGORITHMES DE FOURMIS PARETO POUR RÉSOUDRE UN PROBLÈME D'ORDONNANCEMENT REPRÉSENTATIF DE CONTEXTES RÉELS

2015

RÉSUMÉ

L'ordonnancement multi-objectifs est un domaine de recherche fertile. Les entreprises de tous les secteurs d'activités sont appelées à résoudre des problèmes complexes, de grandes dimensions, souvent de nature combinatoire et généralement à plusieurs objectifs contradictoires ou non commensurables. Ces entreprises doivent alors être en mesure de traiter avec des problèmes multi-objectifs (PMO). Dans le domaine de l'ordonnancement, la machine unique est un problème classique. Toutefois, les hypothèses simplificatrices émises en théorie créent un écart entre la pratique et la théorie. Un des écarts régulièrement présent est la considération du temps de réglage entre les tâches indépendant de la séquence. Dit autrement, le temps de réglage est considéré comme négligeable en théorie. Cependant, cette simplification n'est pas représentative de la plupart des contextes réels. Différentes études présentent l'impact des temps de réglage sur les systèmes de production industrielle. La conséquence de minimiser les temps de réglage est l'amélioration des délais de livraison (Conner 2009; Panwalkar et al. 1973). Pour réduire les écarts entre la pratique et la théorie, plusieurs PMO traitant de la machine unique ont proposé d'ajouter des caractéristiques pour réduire cet écart, dont la machine unique multi-objectifs avec fenêtre d'échéance d'Arroyo et al (Arroyo et al. 2011) (MURMO).

La résolution d'un PMO consiste à produire un ensemble de solutions de compromis entre les objectifs avec ou sans l'aide d'un décideur. Talbi (2009) propose une classification des méthodes de résolution qui comprend, par exemple, les approches Pareto. Ces dernières utilisent la dominance au sens Pareto (Edgeworth 1881; Pareto 1896) pour évaluer la qualité des solutions. Seules les solutions non-dominées sont conservées dans l'ensemble de solutions de compromis, aussi appelé ensemble de solutions Pareto-Optimal (PO).

Parmi les méthodes de résolutions, les algorithmes évolutionnaires (AE) connaissent de bons résultats en uni-objectif. Schaffer (1985) et Goldberg (1989) proposent les premières méthodes de résolution multi-objectifs utilisant les AE. Plusieurs AE ont été adaptés pour résoudre un PMO, dont les plus grandes réussites: NSGA-II (Deb et al. 2000), PMS^{MO} (Zinflou et al. 2008) et GISMOO (Zinflou et al. 2012). NSGA-II pose les fondements des AE Pareto en introduisant le concept d'assignation de performance. Cette assignation évalue la qualité d'une solution par rapport aux autres selon des facteurs de dominance et d'isolement. Le facteur de dominance évalue à quel point la solution domine les autres solutions d'un ensemble. Le facteur d'isolement évalue la densité des solutions qui entourent la solution estimée. Ce mémoire présente l'adaptation de trois AE Pareto de la littérature au problème MURMO. Cette adaptation permet de créer une banque de résultats de comparaison. Les AE Pareto sont, par la suite, comparés avec le seul algorithme connu qui résout le problème MURMO, le MOVNS3 (Arroyo et al. 2011).

La littérature dénombre d'autres types de méthodes de résolution basées sur la construction d'une population de solutions plutôt que sur l'évolution de la population au travers de générations, tel que vu en AE. Il y a entre autres les algorithmes appartenant à la famille de l'optimisation par colonie de fourmi (OCF) qui connaissent aussi de bons résultats en uni-objectif et leur utilisation pour résoudre les PMO ne fait qu'augmenter. La revue de la littérature des algorithmes d'OCF Pareto démontre que peu d'algorithmes sont basés sur l' « ant colony system » (ACS) et que le nombre de colonies a un impact sur la conception de la méthode de résolution. Ce mémoire propose la comparaison entre une méthode multi-colonies et une méthode uni-colonie. Également, cette proposition démontre l'intérêt d'emprunter des concepts appartenant traditionnellement aux algorithmes évolutionnaires pour les adapter à d'autres algorithmes. La méthode multi-colonie est une adaptation pour le problème MURMO de l'algorithme proposé par Iredi et al (2001). Cet algorithme n'utilise pas d'assignation de performance des AE Pareto. L'algorithme d'OCF est basé sur un AS. Pour sa part, la méthode uni-colonie est représentée par la transposition de l'algorithme « genetic immune system for multiple objective optimization » (GISMOO) (Zinflou et al. 2012) vers un algorithme ACS. Cette transposition permet d'inclure l'assignation de performance dans l'algorithme d'OCF. Une comparaison équitable est proposée entre les méthodes multi-colonies et uni-colonie. Pour terminer, le mémoire présente une bonification de la transposition. Cette bonification a pour ambition d'améliorer les résultats de la méthode uni-colonie.

REMERCIEMENTS

Tout d'abord, je désire remercier Mme Caroline Gagné, ma directrice de recherche, pour sa disponibilité et son support. J'aimerais souligner son engagement et son optimisme envers moi et ma recherche, ce qui a permis à ce mémoire d'atteindre la qualité souhaitée.

Je tiens également à remercier tous les membres du Groupe de Recherche en Informatique de l'UQAC (GRI) pour l'ambiance de travail qui réside dans les bureaux. Par la même occasion, je désire remercier mes collègues et amis avec qui j'ai partagé ce bureau, Hugo Deschênes et Mathieu Fournier, pour leur expertise mathématique et leur support.

Je remercie aussi Messieurs Aymen Sioud et Arnaud Zinflou pour avoir accepté d'évaluer ce travail.

Des études supérieures sont des projets qui touchent toutes les sphères de nos vies. Pour cette raison, je tiens à remercier mon conjoint et ami Jérémy Beauregard. Sans son soutien sans faille, sa foi en moi et son implication, les études supérieures n'auraient pas été possibles. Je tiens particulièrement à souligner sa contribution à améliorer la qualité du français de ce document. Dans la même optique, je désire remercier ma mère, Rolande Côté, ma sœur, Chantale LeBel et mon filleul, Jonathan Bergeron pour leur encouragement et leurs diners qui m'ont apporté réconfort et détente.

Pour terminer, je tiens à remercier mes amis(es) qui ont été une source de détente et de ressourcement tout le long de la maîtrise.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1

INTRODUCTION

Les entreprises de tous les secteurs d'activités sont appelées à résoudre des problèmes complexes, de grande dimension et souvent de nature combinatoire. De plus, ces problèmes ont, généralement, des objectifs contradictoires ou non commensurables. Ces entreprises sont donc concernées par l'optimisation combinatoire multi-objectifs.

L'optimisation multi-objective a été étudiée pour la première fois au XIX^{e} siècle dans les travaux d'Edgeworth(1881) et de Pareto(1896). Cette science fit ses preuves, en premier lieu, dans le domaine de l'économie ainsi que dans la gestion et graduellement dans le domaine informatique.

La plupart des problèmes d'optimisation combinatoire ont un ou plusieurs objectifs et sont difficiles au sens de la théorie de la complexité. En effet, de nombreux problèmes de la littérature tels que le problème du voyageur de commerce, le sac alpin, l'affectation quadratique ou l'ordonnancement d'une machine unique avec minimisation du retard, pour nommer que ceux-ci, sont NP-Difficiles (Du et al. 1990; Garey et al. 1979). La résolution d'un problème d'optimisation se résume en la détermination de la « meilleure solution » assujettie à un ensemble de contraintes. La mesure de performance détermine si une solution est meilleure qu'une autre et est appelée fonction objectif. Le but de la résolution d'un problème d'optimisation est donc de trouver la solution réalisable ayant la fonction objectif minimale (ou maximale), aussi appelée solution optimale.

Dans le cadre des problèmes multi-objectifs (PMO), la qualité de solution est plus difficile à définir en raison de la pluralité des objectifs. Cette pluralité a comme conséquence qu'il n'existe pas une solution, mais un ensemble de solutions, que l'on appellera ensemble de solutions Pareto optimales (PO). Résoudre un PMO se réalise

généralement en deux étapes : la première consiste à trouver l'ensemble des solutions de compromis avec ou sans l'aide d'un décideur et la seconde fait intervenir le décideur pour choisir la solution la plus appropriée.

Tel que décrit par Talbi (1999; 2009), la conception d'une méthode de résolution d'un PMO est difficile, car il n'y a pas de définition de l'optimalité comme dans un problème uni-objectif. En effet, pour un PMO, la relation d'ordre entre les solutions PO est partielle. Le choix final revient à un décideur en fonction de ses préférences. L'ensemble PO augmente en fonction de la taille du problème et du nombre d'objectifs. Les ensembles PO ont différentes structures : convexe/concave, continue/discontinue ou multi-modale (Talbi 1999). Ces structures dépendent du PMO.

Différentes classifications des méthodes de résolution de PMO ont été proposées (Talbi 2009), la classification utilisée dans ce mémoire est celle *concepteur*. Talbi (1999) regroupe la classification concepteur en trois catégories : les approches basées sur la transformation du problème en uni-objectif, les approches non-Pareto et les approches Pareto. La transformation en uni-objectif s'articule, généralement, autour d'un vecteur donné par le décideur. Dans les approches non Pareto, la recherche de solutions s'effectue en traitant les objectifs séparément (Talbi 1999). Pour leur part, les approches Pareto différencient les solutions à l'aide de la notion de dominance au sens Pareto. La dominance au sens Pareto est une relation partielle entre les solutions. Ce mémoire s'intéresse aux approches Pareto. Les notions touchant les approches Pareto seront détaillées à la Section 2.4.

Les algorithmes exacts peuvent parfois résoudre des PMO. Toutefois, leurs temps d'exécution imposants sur des problèmes de grande envergure les rendent souvent inutilisables. Le besoin de trouver une réponse rapidement à un PMO devient plus important que d'avoir une solution exacte. L'utilisation de méthodes approchées telles que les métaheuristiques est alors indiquée (Dhaenens-Flipo 2005).

Dans le but de résoudre des problèmes multi-objectifs posés par Schaffer (1985), Goldberg (1989) propose l'utilisation de la dominance au sens Pareto dans les algorithmes évolutionnaires (AE), particulièrement l'algorithme génétique (AG). Tel que souligné par Francisci (2002), l'AG a la possibilité d'exploiter un vaste espace de recherche et de générer plusieurs solutions à chaque itération. Les AGs sont devenus très populaire pour résoudre des PMO avec une approche Pareto. Sans être exhaustives, plusieurs recherches ont été réalisées à ce sujet (Deb et al. 2002; Fujita et al. 1998; Govindarajan 2005; Kumar 2002; Li 2009; Obayashi et al. 1998; Oliveira 2003; Tamura 2007; Zhou et al. 1999; Zinflou 2008; Zinflou et al. 2012).

La famille des métaheuristiques d'optimisation par colonie de fourmis (OCF) comporte l'ensemble des approches s'inspirant de la synergie entre les fourmis. La première version proposée est le ant system (AS) (Colorni et al. 1992). Des versions plus évoluées ont par la suite été proposées dans la littérature, entre autres celle de Dorigo et Gambardella (1997) le ant colony system (ACS), qui est l'algorithme de base pour ce mémoire (Bullnheimer et al. 1997; Dorigo et al. 1997; Dorigo et al. 1996; Gambardella et al. 1995; Maniezzo 1999; Stützle et al. 2000). L'intérêt de la métaheuristique ACS s'explique, car elle permet d'exploiter un vaste espace de recherche par ses paramètres liés

à l'importance de la piste de phéromones et de la visibilité. De plus, cette métaheuristique a la possibilité de générer une multitude de solutions à chaque itération, car elle est à base de population (colonie) où chaque individu (fourmi) représente une solution.

L'ACS est une bonne candidate pour les PMO, car elle offre des caractéristiques similaires à un AG, tel que générer plusieurs solutions lors d'une itération. La littérature démontre que les ACS générant un ensemble PO avec une approche Pareto sont peu courants (Dorigo et al. 2004; Moncayo-Martinez et al. 2011). L'ensemble des méthodes ACS Pareto peut être divisé en deux catégories, soit celles utilisant une seule colonie (uni-colonie) et celles utilisant plusieurs colonies (multi-colonies). Lors de la création d'une méthode de résolution ACS Pareto, les deux catégories entraînent des modifications importantes à l'algorithme initial de Dorigo et Gambardella (1997). L'évaluation de la qualité des solutions est un des exemples de modifications. Les algorithmes uni-colonie évaluent la qualité de chaque fourmi individuellement. Quant aux algorithmes multi-colonies, l'évaluation de la qualité est effectuée seulement si l'algorithme utilise des concepts reconnus par les AE tels que le facteur de dominance ou le facteur d'isolement. Dans le cas où de tels concepts ne sont pas utilisés, l'évaluation de la qualité est basée sur la dominance d'une fourmi par rapport aux autres. L'évaluation est utilisée essentiellement dans la mise à jour d'une archive globale de solutions non-dominées. L'utilisation d'une ou de plusieurs colonies fait partie des choix à poser dans la mise en œuvre d'un ACS Pareto.

Ce mémoire consiste à adapter deux algorithmes de la littérature, soit un algorithme uni-colonie et l'autre multi-colonie. Par la suite, ces algorithmes sont exécutés et comparés sur un problème multi-objectifs d'ordonnancement prouvé NP-difficile intitulé problème

multi-objectifs de machine unique avec réglage et fenêtre d'échéance (MURMO) (Arroyo et al. 2011).

Le présent mémoire est subdivisé comme suit : dans un premier temps, le Chapitre 2 détaille les concepts énumérés dans l'introduction ainsi que différentes méthodes de résolution de PMO utilisant les principes d'AE et d'OCF. Dans un second temps, le Chapitre 3 oriente le discours vers la méthode choisie pour ce mémoire : l'ACS Pareto. Ce chapitre présente tout d'abord une revue de la littérature des algorithmes multi-colonies et uni-colonie. Ensuite, les méthodes de résolution ayant été proposées pour résoudre MURMO sont présentées. L'univers d'expérimentation numérique ainsi que les résultats et leur analyse sont présentés dans un troisième temps au Chapitre 4. L'analyse conduit finalement à la conclusion du mémoire ainsi qu'à une ouverture sur les recherches futures au Chapitre 5.

CHAPITRE 2

OPTMISATION MULTI-OBJECTIFS ET MÉTAHEURISTIQUES PARETO

2.1. Introduction

Ce chapitre présente les concepts, notions et classifications de la littérature concernant les PMO ainsi que diverses méthodes de résolution de PMO utilisant les principes d'AE et d'OCF. Cette présentation s'articule autour de la Section 2.2 qui donne une définition formelle d'un PMO. Pour solutionner ces problèmes, la Section 2.3 présente les classifications proposées dans la littérature pour regrouper les méthodes de résolution des PMO. Cette section permet aussi de positionner la recherche dans les approches Pareto. Ces dernières doivent atteindre deux buts : qualité et distribution dans la génération d'un ensemble PO (Talbi 2009). La solution doit *converger* vers l'ensemble PO pour satisfaire le but qualité : ce but dépend de la notion de dominance au sens Pareto. Pour sa part, la *diversification* d'une solution permet aux solutions de couvrir le front Pareto et satisfaire le but de distribution. Pour une meilleure compréhension, la Section 2.4 introduit les notions de dominance et d'optimum Pareto. La Section 2.5 présente les principaux AE Pareto proposés dans la littérature et qui sont parmi les plus beaux succès des métaheuristiques pour la résolution de PMO (Deb et al. 2000; Zinflou et al. 2012; Zinflou et al. 2008). Par la suite, la Section 2.6 présente les approches d'OCF proposées dans la littérature pour résoudre les PMO. Étant donné la pluralité des objectifs en PMO, l'évaluation des ensembles de solutions PO se réalise à partir de métriques qui sont présentées à la Section 2.7. Pour terminer, la Section 2.8 expose les objectifs de la recherche et la méthodologie utilisée.

2.2. Problème multi-objectifs (PMO)

Tel que présenté dans l'introduction, l'optimisation multi-objectifs permet de résoudre des PMO rencontrés dans des contextes réels. Formellement, un PMO dans un contexte de minimisation se définit comme suit :

$$PMO \quad \begin{cases} \min F(x) = \big(f_1(x), f_2(x), \dots, f_n(x)\big) \\ s.c. \;\; x \in S \end{cases} \tag{2.1}$$

où $n \geq 2$ est le nombre de fonctions objectif, $x = (x_1, x_2, \dots, x_m)$ sont les m variables de décision, S représente les solutions réalisables associées à des contraintes d'égalité, d'inégalité ainsi qu'à des bornes et $F(x) = f_1(x), f_2(x), \dots, f_n(x)$ sont les fonctions objectif à minimiser. Résoudre un PMO consiste à déterminer non pas une solution visant l'optimalité, mais plutôt un ensemble de solutions PO. Chaque solution de l'ensemble PO est une solution de compromis entre les objectifs. De plus, ces solutions ne peuvent être discriminées entre elles sans les préférences du décideur pour chaque objectif. Dans cette optique, il existe plusieurs méthodes pour résoudre un PMO, la section suivante les classifie et les détaille.

2.3. Classification des méthodes de résolution pour PMO

De nombreuses recherches classifient les méthodes de résolution pour PMO. Deux façons se distinguent de la littérature : l'interaction avec le décideur, appelée classification décideur, et le traitement apporté aux fonctions objectif, appelé classification concepteur.

2.3.1 Classification décideur

Tel que décrit par Hwang et Masud (1980), il est possible de classifier les approches pour les PMO selon le moment où le décideur exprime ses préférences : a priori, a posteriori et interactive. Les approches *a priori* exigent que l'information sur les préférences du décideur soit connue avant la résolution du problème afin de guider la méthode de résolution. Dans ce cas, le décideur doit connaitre le problème ainsi que les facteurs qui s'y rattachent. En ce qui concerne les approches *a posteriori*, la résolution du problème se fera sans aucune information sur les préférences du décideur. Les approches a posteriori donnent ainsi un ensemble PO large au décideur, qui choisit alors une solution qui satisfait au mieux ses préférences. Cette approche est exploitable exclusivement quand la cardinalité de l'ensemble PO est réduite (Sayin et al. 1999). Les approches *interactives* permettent une coopération progressive entre le décideur et la méthode. Cette coopération s'articule au travers des itérations; avec les connaissances acquises de l'ensemble PO de l'itération précédente, le décideur exprime ses préférences qui sont prises en considération dans l'itération suivante. Cette méthode ne garantit pas de trouver la solution finale en un nombre fini d'itérations.

2.3.2 Classification concepteur

Les approches de la classification concepteur (Talbi 2009) sont différenciés entre elles par le traitement apporté aux fonctions objectif. Il y a trois traitements répertoriés : les approches basées sur la transformation des fonctions objectif en uni-objectif, les approches non-Pareto et les approches Pareto.

La *transformation des fonctions objectif en uni-objectif* fait intervenir le décideur en début de méthode pour donner ses préférences sur les fonctions objectif à optimiser. La préférence du décideur sur les fonctions objectif prend généralement la forme d'un vecteur (Berro 2001) où chaque indice représente une fonction objectif. Le vecteur peut contenir le poids qu'a une fonction objectif par rapport aux autres : on parle alors d'agrégation (Hwang et al. 1980). Dans ce cas-ci, chaque indice du vecteur est une variable multiplicative pondérant la fonction objectif. Le vecteur peut aussi représenter les buts à atteindre pour chaque fonction objectif : on parle alors de programmation par but (Charnes et al. 1961). Pour cette dernière méthode, le défi est de minimiser l'écart entre le but et la fonction objectif pour un objectif donné.

Dans les approches *non Pareto*, les fonctions objectif sont traitées séparément (Talbi 2009). Parmi les principales approches, nous retrouvons la sélection lexicographique où le décideur donne l'ordre des fonctions objectif à prioriser. Le « Vector Evaluated Genetic Algorithm » (VEGA) (Schaffer 1985) effectue la sélection des individus de la population courante suivant chaque objectif.

Les approches *Pareto* sont de type a posteriori. Elles différencient les fonctions objectif à l'aide de la notion de dominance au sens Pareto. La dominance est une relation d'ordre partiel entre les fonctions objectif des solutions. En conséquence, les approches Pareto sont retenues dans le cadre de ce mémoire.

2.4. Notion de dominance et optimum Pareto

Au XIX^{e} siècle, un mathématicien italien, Vilfredo Pareto, formule le concept suivant : dans un ensemble Pareto Optimal d'un PMO, il existe un équilibre tel que l'on ne peut pas améliorer un objectif sans détériorer au moins un des autres objectifs (Pareto 1896). Cet équilibre est appelé Pareto Optimal (PO).

Dans le cadre d'un PMO, les solutions réalisables sont ramenées dans l'espace objectif. Une relation d'ordre partiel, appelée dominance au sens Pareto, est établie entre les solutions de l'espace objectif. Une solution $x \in S$ est dite PO si elle n'est dominée par aucune autre solution de l'espace objectif. Autrement dit, dans un contexte de minimisation, une solution $x \in S$ *domine* une solution $y \in S$ tel que $f_i(x) \leq f_i(y)$ pour tout $i = 1, \dots, n$ et qu'il existe au moins un j tel que $f_j(x) < f_j(y)$. Une solution PO est aussi appelée solution efficace, non-dominée ou non inférieure. L'ensemble des solutions PO forme le front Pareto. Deux solutions particulières de l'espace objectif donnent de l'information sur la portée du front Pareto : les solutions idéale et nadir. La solution *idéale* est une solution utopique, qui n'appartient généralement pas à l'ensemble de solutions réalisables et qui optimise chacune des fonctions objectif individuellement. La solution *nadir* est l'opposée de la solution idéale.

Les notions Pareto permettent d'évaluer une solution selon sa dominance par rapport à une autre. Il est alors possible de dire qu'une solution domine une autre solution. Mais quelle est la relation de la dominance, est-elle forte ou faible? Zitzler et al. (2003) proposent des relations de dominance incluant un ensemble d'annotations. Ces relations permettent de déterminer quel type de dominance il y a entre deux solutions ou deux

ensembles PO : domine strictement, domine, meilleur, domine faiblement ou incomparables. Le Tableau 1 énumère les relations pour les solutions et les ensembles PO.

Relation	**Solution**		**Ensemble PO approximé**	
Domine strictement	$z^1 \succ\succ z^2$	z^1 est meilleure que z^2 sur tous les objectifs	$A \succ\succ B$	Tout $z^2 \in B$ est strictement dominé par au moins un $z^1 \in A$
Domine	$z^1 \succ z^2$	z^1n'est pas pire que z^2 pour tous les objectifs et meilleure sur au moins un objectif	$A \succ B$	Tout $z^2 \in B$ est dominé par au moins un $z^1 \in A$
Meilleur	N/A	N/A	$A \rhd B$	Tout $z^2 \in B$ est faiblement dominé par au moins un $z^1 \in A$ et $A \neq B$
Domine faiblement	$z^1 \succcurlyeq z^2$	z^1 n'est pas pire que z^2 pour tous les objectifs	$A \succcurlyeq B$	Tout $z^2 \in B$ est faiblement dominé par au moins un $z^1 \in A$
Incomparable	$z^1 \parallel z^2$	Ni z^1 ne domine faiblement z^2, ni z^2 ne domine faiblement z^1	$A \parallel B$	Ni A ne domine faiblement B, ni B ne domine faiblement A

Tableau 1 : Relations de dominance pour les solutions et les ensembles PO (Zitzler et al. 2003)

La première colonne du Tableau 1 représente les différents types de relation. La deuxième colonne donne une description de la relation pour deux solutions : z^1 et z^2. Enfin, la dernière colonne donne une description de la relation pour deux ensembles PO approximés : A et B. Les relations présentées ici sont généralement utilisées pour comparer la qualité des solutions de deux ensembles PO. Également, ces relations servent à définir des métriques.

Pour générer des solutions PO dans toute la région du front Pareto, la méthode doit atteindre deux buts : la qualité et la distribution. La *qualité* des solutions représente le

principe que les solutions doivent converger vers le front Pareto. Une solution de bonne qualité se retrouve près ou sur le front Pareto et à l'inverse, un ensemble des solutions de qualité médiocre est éloignée du front Pareto. La *distribution* représente le principe que les solutions trouvées doivent se situer tout le long du front Pareto et non seulement dans un seul secteur. Une bonne distribution est caractérisée par une diversité des solutions sur le front Pareto et une mauvaise distribution a pour effet que toutes les solutions seront regroupées dans un secteur du front Pareto. Ces caractéristiques permettent d'évaluer un ensemble PO de façon qualitative. La Section 2.7 présente des méthodes quantitatives d'évaluation d'ensembles PO, soit les métriques. La prochaine section présente les algorithmes métaheuristiques les plus réputés de la littérature en optimisation multi-objectifs.

2.5. Algorithme évolutionnaire (AE) Pareto

Les AE sont traités dans le cadre de ce mémoire, car ils sont parmi les méthodes les plus performantes pour résoudre tant des problèmes uni-objectif que des PMO. Ces méthodes placent également les bases de l'optimisation multi-objectifs utilisant les métaheuristiques. Cette section présente une revue de la littérature des principales méthodes AE qui servent de comparaison et de base de réflexion pour ce mémoire.

Les AE en optimisation uni-objectif s'inspirent de la théorie de l'évolution de Charles Darwin en adoptant autant les concepts que la terminologie. Ces méthodes sont basées sur l'évolution d'un ensemble d'individus (population) dans le temps (génération). Dans un AE uni-objectif, une population initiale d'individu (solution) est formée en début de processus. Cette population évolue au travers des générations. Pour ce faire, deux individus sont

sélectionnés pour le croisement, en privilégiant les meilleurs individus. Lors du croisement, un nouvel individu (enfant) est créé présentant des caractéristiques de ses deux parents. Pour représenter l'évolution quelques fois chaotique en contexte réel, une mutation, soit une légère modification, est appliquée à certains enfants. Suite à la création des solutions enfants, une nouvelle sélection est réalisée afin de former la prochaine génération. La sélection est faite au travers des populations parent et enfant en privilégiant les meilleures. Cette phase a pour analogie que les plus forts des générations survivent.

Les AE ont également été adaptés pour résoudre des PMO. De nombreux AE sont proposés dans la littérature et il est possible de regrouper ces algorithmes selon la classification concepteur précédemment abordée. Il est à noter que cette présentation se concentre uniquement sur les AE Pareto qui servent de base de réflexion dans la suite de ce mémoire. Une des principales difficultés dans la résolution de PMO par un AE Pareto concerne l'établissement d'une mesure de qualité permettant d'assigner une relation d'ordre entre les solutions dans la population. Étant donné que la qualité d'une solution dépend de l'évaluation de plusieurs fonctions objectif contradictoires et souvent non commensurables, un nouveau concept est donc introduit par les EA Pareto : l'*assignation de la performance*. Ce concept permet de traduire la qualité d'une solution en fonction de deux facteurs : le *facteur de dominance* et le *facteur d'isolement*. Le premier facteur permet de mesurer le degré de dominance d'une solution au sens Pareto et le deuxième permet d'évaluer la densité de solutions qui entourent une solution donnée. La suite de cette section présente trois AE pour résoudre des PMO reconnus pour leur performance : NSGA-II, PMS^{MO} et GISMOO. Les AE sont comparés principalement sur leur assignation de performance.

2.5.1. Non-dominated Sorting Genetic Algorith II (NSGA-II)

Le NSGA-II a été proposé par Deb et al. (2000). Cette méthode est la référence en AE Pareto. Comme son nom l'indique, il s'agit de la deuxième version de l'algorithme proposée par les auteurs. Le NSGA-II permet de pallier à diverses critiques formulées à l'endroit de la première version telles que le non-élitisme et l'utilisation d'un paramètre pour maintenir la diversité. Le pseudo-code du NSGA-II est présenté à la Figure 1.

1 :	Affecté l'itération courante t à 0
2 :	Initialiser la population P_t de grandeur M
3 :	Q_t = représente une nouvelle population par croisement et mutation à partir de P_t
4 :	Tant que $t <$ le nombre de générations maximales
5 :	$U_t = P_t \cup Q_t$
6 :	$D =$ trier par front (U_t)
7 :	$P_{t+1} = \emptyset$ et $i = 1$
8 :	Tant que $P_{t+1} \leq M$
9 :	Calculer l'encombrement de D_i
10 :	$P_{t+1} = P_{t+1} \cup D_i$
11 :	$i = i + 1$
12 :	Tirer D_i en ordre de dominance avec discrimination sur la plus grande distance
13 :	$P_{t+1} = P_{t+1} \cup U_i[M - P_{t+1}]$
14 :	$Q_{t+1} =$ faire une nouvelle population par croisement et mutation à partir de P_{t+1}
15 :	$t = t + 1$

Figure 1 : Illustration du Pseudo-code du NSGA-II (Deb et al. 2000)

Le NSGA-II commence par initialiser sa population parent P_t ainsi qu'une nouvelle population enfant Q_t par croisement et mutation à partir de la population P_t. Les lignes 4 à 15 de la Figure 1 représentent une génération. La première étape est de trier par front la population U_t où U_t est la population engendrée par l'union de la population parent à la génération t (P_t) et la population enfant à la génération t (Q_t). NSGA-II introduit le « Fast

non-dominated sort », le tri par front d'une population selon la dominance de ses individus. Les solutions du premier front sont non-dominées, les solutions du deuxième front sont dominées seulement par celles appartenant au 1er front, et ainsi de suite. Ce tri est régulièrement réutilisé dans les méthodes d'optimisation multi-objectives. L'étape suivante se situe entre les lignes 8 à 13 de la Figure 1. Cette étape crée une nouvelle population parent P_{t+1} à partir des solutions du premier front (D_1) calculées plus tôt. Pour ce faire chaque solution de D_1 reçoit un encombrement (crowded comparison) appelé assignation de performance dans ce mémoire. L'assignation de performance du NSGA-II s'effectue selon le facteur de dominance et le facteur d'isolement. Le facteur de dominance est représenté par le numéro du front que la solution a reçu suite au « Fast non-dominated sort ». Le facteur d'isolement est désigné par l'estimation de la densité autour de la solution (crowdin distance). Dit autrement, elle représente une normalisation de la distance euclidienne. Les facteurs sont utilisés lors de la sélection pour le croisement (ligne 14) et la sélection de la prochaine génération (lignes 8 à 13). Quand chaque solution du premier front possède son assignation de performance, alors les solutions sont triées (ligne 12) en ordre du facteur de dominance avec discrimination sur le facteur d'isolement en cas d'égalité. Le résultat de ce tri sert à remplir la population parent P_{t+1} de la prochaine génération. Pour terminer la génération actuelle (t), une nouvelle population enfant Q_{t+1} est générée par croisement et mutation de la population parent P_{t+1} (ligne 14).

2.5.2. Pareto Memetic Strategy for Multiple Objective optimization (PMS^{MO})

Le PMS^{MO}, proposé par Zinflou et al. (2008), est une hybridation entre l'algorithme génétique et la recherche locale. L'originalité du PMS^{MO} provient de son assignation de performance qui considère l'historique des solutions trouvées. Le pseudo-code du PMS^{MO} est présenté à la Figure 2:

1 : Initialiser l'archive locale A_0 et l'archive globale $\widetilde{A_0}$ à $\emptyset$
2 : Initialiser aléatoirement la population P_0 de grandeur M
3 : Évaluer chaque individu $x \in P_0$
4 : Calculer l'assignation de performance de chaque individu $x \in P_0$
5 : Mise à jour de A_0 avec les individus non-dominés $x \in P_0$
6 : Trier P_0 sur la dominance des individus $\in P_0$ avec le tri par front du NSGA-II
7 : Tant que $t <$ le nombre de générations maximales
8 : Sélectionner des individus dans P_t et créer Q_t par croisement et mutation
9 : Évaluer chaque individu $x \in Q_t$
10 : Effectuer une amélioration locale sur chaque individu $x \in Q_t$
11 : Mise à jour de A_t avec les individus non-dominés $\in Q_t$
12 : $U_t = P_t \cup Q_t$
13 : Calculer l'assignation de performance de chaque individu $x \in U_t$ et $y \in A_t$
14 : Si $A_t.size > M$ alors
15 : Réduire A_t et mise à jour de $\tilde{A}_t$
16 : Trier U_t sur la dominance des individus $\in U_t$ avec le tri par front du NSGA-II
17 : Copier les M premières solutions de U_t dans P_{t+1}

Figure 2 : Pseudo-code du PMS^{MO} (Zinflou et al. 2008)

L'assignation de performance est dirigée par les facteurs de dominance et d'isolement tel qu'expliqué précédemment, mais ils diffèrent dans leur opérationnalisation. Le facteur de dominance du PMS^{MO} est calculé en tenant compte du nombre d'individus dominés par la solution *x* et du nombre d'individus dominant la solution *x* où x appartient à l'archive locale A ou à la population parent P. Cette approche diffère du SPEA2 (Zitzler et al. 2001)

dans le sens que le rang donné à un individu ne dépend pas seulement de la force (strength) de celui-ci, mais aussi de la dominance qu'il a sur les autres individus (Zinflou et al. 2008). Le facteur d'isolement, emprunté au SPEA2 (Zitzler et al. 2001), est basé sur la densité de solutions. Les facteurs sont utilisés dans l'algorithme lors de la sélection d'individus pour le croisement à la ligne 8 de la Figure 2. La sélection de la ligne 8 est un tournoi binaire basé sur le facteur de dominance où le plus petit facteur de dominance l'emporte avec discrimination sur le plus grand facteur d'isolement en cas d'égalité. La gestion de l'élitisme du PMS^{MO} s'effectue à l'aide de deux archives de solutions non-dominées. La première archive A est locale et limitée. L'archive globale $\tilde{A}$ n'est pas de taille constante et elle est illimitée.

2.5.3. Genetic Immune Strategy for Multiple Objective Optimization (GISMOO)

La caractéristique principale de GISMOO, proposé par Zinflou et al. (2012), est l'hybridation entre un algorithme génétique et un système immunitaire artificiel. La particularité de cette hybridation est qu'elle est réalisée lors de la phase de la création de la population enfant. Celle-ci est générée à 50% par croisement et 50% par une hypermutation empruntée au système immunitaire artificiel, tel que le démontre la Figure 3.

1 : Initialiser la population parent P_0 de grandeur M
2 : Initialiser l'archive A et la population enfant Q à $\emptyset$
3 : Évaluer chaque solution $x \in P_0$
4 : Mise à jour de A avec les individus non-dominés $\in P_0$
5 : Calculer l'assignation de performance de chaque individu $x \in P_0$
6 : $t \leftarrow 1$
7 : Tant que $t <$ le nombre de générations maximales
8 : Tant que $Q_t.size < {}^{M}/_{2}$
9 : Sélectionner deux individus π_1 et $\pi_2 \in P_t$ par tournoi binaire
10 : Créer deux enfants enf_1 et enf_2 par croisement de π_1 et π_2
11 : Évaluer enf_1 et enf_2 et effectuer la mise à jour de A
12 : Sélectionner aléatoirement un nombre $aléa \in [0,1]$
13 : Si $aléa <$ probabilité de mutation
14 : Effectuer la mutation sur enf_1 et enf_2
15 : Winner = compétition (enf_1, enf_2)
16 : Ajouter Winner dans Q_t
17 : Trier P_t par front → D
18 : Pour chaque solution $x \in$ au premier front (D_1) calculer le $nbClone$ avec l'équation 2.2
19 : Cpt ← 0
20 : Tant que $cpt < nbClone(x)$
21 : Générer deux clones cl_1 et cl_2 de x
22 : Créer cl_1^{mut} par mutation1 sur cl_1
23 : Créer cl_2^{mut} par mutation2 sur cl_2
24 : Évaluer cl_1^{mut} et cl_2^{mut}, mise à jour de A
25 : Winner = compétition(cl_1^{mut}, cl_2^{mut})
26 : Ajouter Winner dans Q_t
27 : Cpt ++
28 : Calculer l'assignation de performance de chaque solution $x \in P_t \cup Q_t$
29 : Trier $P_t \cup Q_t$
30 : Copier les M premières solutions de $P_t \cup Q_t$ dans P_{t+1}
31 : $t++$
32 : Retourner A

Figure 3: Pseudo-code GISMOO (Zinflou et al. 2012)

La phase génétique, située entre de la ligne 8 à 16 à la Figure 3, est empruntée aux AE, à l'exception de la méthode *compétition* qui détermine entre deux individus lequel est le meilleur. Le pseudo-code de *compétition* est présenté à la Figure 4.

```
1 :   Si x domine y alors
2 :     Retourne x
3 :   Sinon
4 :     Si x est dominé par y alors
5 :       Retourne y
6 :     Sinon
7 :       Si facteurIsolement(x) > facteurIsolement(y) alors
8 :         Retourne x
9 :       Sinon
10 :        Si facteurIsolement(x) < facteurIsolement(y) alors
11 :          Retourne y
12 :        Sinon
13 :          Retourne un choix aléatoire entre x et y
```

Figure 4 : Pseudo-code de compétition (individu x, individu y) (Zinflou 2008)

La *compétition* est remportée par l'individu qui domine l'autre tel que le démontrent les lignes 1 à 5. Au cas où les deux individus ne seraient pas dominés entre eux, l'individu ayant le plus grand facteur d'isolement l'emporte (lignes 7 à 11). Dans l'éventualité d'une égalité parfaite, alors le choix se fait aléatoirement.

Pour la phase immune, située entre les lignes 18 et 27 à la Figure 3, le nombre de clones produits à partir d'une solution (anticorps) est en fonction de son facteur d'isolement. Plus l'anticorps est éloigné et plus il va générer des clones. Le nombre de clones est calculé en utilisant l'équation suivante :

$$nbClone(x) \leftarrow round\left[\left(Dist(x) * {}^{M}/_{2}\right)\Big/ \sum_{y=1}^{D_1.size} Dist(y)\right] \qquad (2.2)$$

où $Dist(x)$ représente le facteur d'isolement de l'anticorps x et M le nombre de solutions maximales de la population. Dans le présent cas, les anticorps correspondent aux individus non-dominés de la population courante P_t. L'assignation de performance est dirigée selon

les mêmes facteurs rencontrés dans les algorithmes évolutionnaires multi-objectifs, soit la dominance et l'isolement. Le facteur de dominance est emprunté au PMS^{MO} en utilisant une population différente, soit la population parent P combinée à la population enfant Q. Le facteur d'isolement provient de la métrique d'Espacement de Schott (1995) qui évalue la distance séparant un individu *x* de ses plus proches voisins. Les facteurs sont, entre autres, utilisés dans la phase génétique. Le facteur d'isolement permet de différencier deux individus lors de la sélection de la prochaine population P_{t+1} ainsi que lors du tri de la population. En cas d'égalité, il y a discrimination sur le facteur de dominance. Pour la phase immune, le facteur d'isolement détermine le nombre de clones à produire pour un individu *x* donné. Si l'individu est isolé, alors il génère plus de clones.

Les AE Pareto vus dans les dernières sections posent les balises de la résolution de PMO avec l'assignation de performance qui permet d'évaluer les solutions en utilisant les principes de dominance au sens Pareto. Les AE sont des méthodes évolutionnaires; autrement dit, elles améliorent une population initiale au travers de générations (itérations). Les méthodes qui intéressent ce mémoire sont plutôt constructives. Elles génèrent à chaque itération de nouvelles solutions. Elles intègrent également dans leur calcul des mécanismes pour conserver une trace des solutions passées. En outre, les approches OCF multi-objectifs empruntent quelques fois les concepts des AE pour traiter les PMO.

2.6. Optimisation par colonie de fourmis Pareto

Les AE Pareto comptent parmi les méthodes les plus performantes de la littérature. Toutefois, les approches de la famille d'OCF connaissent également des bons résultats pour l'optimisation uni-objectif. Cette réussite est occasionnée par leur capacité à exploiter un

vaste espace de recherche et à générer une multitude de solutions à chaque itération. Cette section présente une revue de la littérature des approches d'OCF qui sont le fondement des méthodes présentées dans ce mémoire.

L'observation des fourmis dans le monde réel permet de découvrir qu'elles ont la capacité de trouver le plus court chemin entre la colonie et une source de nourriture. Ceci est possible car chaque fourmi suit les phéromones déposées précédemment par ses collègues (Beckers et al. 1992; Goss et al. 1989). Plus le chemin est court et plus la trace de phéromones est renforcée rapidement. La section suivante présente plusieurs approches OCF : ant systems (AS), min-max ant system (MMAS) et ant colony system (ACS). Pour conceptualiser ces approches, elles sont présentées à l'aide d'un problème répandu dans la littérature : le problème du voyageur de commerce uni-objectif.

2.6.1. Problème de voyageur de commerce (VC) pour des approches OCF

Le problème du VC consiste à trouver le plus court chemin reliant V villes où chaque ville ne doit être visitée qu'une seule fois et ensuite retourner à la ville initiale. Généralement, le problème est formellement défini à l'aide d'un graphe $G(V, L)$ avec V nœuds qui représentent l'ensemble des villes et L qui représente les arcs reliant chaque paire de villes. Pour chaque paire de villes (i, j), une distance l_{ij} est assignée tel que l_{ij} n'est pas forcément égale à l_{ji}. Le but est de trouver une permutation de villes $x = (x_1, x_2, \dots, x_V)$ qui minimise la distance :

$$f(x) = \sum_{i=1}^{V-1} l_{x(i)x(i-1)} + l_{x(V)x(1)} \qquad (2.3)$$

2.6.1.1. AS pour le problème de voyageur de commerce (VC)

L'AS est un algorithme de construction qui reprend la terminologie de la synergie des fourmis expliquée plus haut. À la différence de l'AE qui crée une population initiale et qui la fait évoluer au travers d'itérations, l'AS est dite constructive. Elle crée à chaque itération (cycle) un ensemble de solutions (colonie) qui seront effacées après avoir extrait l'information des solutions produites par un dépôt de phéromone en fonction de la qualité des solutions. Chaque solution (fourmi) est construite progressivement en choisissant une ville disponible. Une ville est disponible si elle n'a pas encore été choisie par la fourmi. Le choix d'une ville par rapport à un autre s'effectue à l'aide d'une règle de choix. La règle de choix a pour base les phéromones (τ) et la visibilité (η). Elle se calcule avec l'Équation 2.4 qui détermine une probabilité p de choisir pour la fourmi k la ville j après la ville i.

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\left(\tau_{ij}(t)\right)^{\delta} * (\eta_{ij})^{\varepsilon}}{\sum_{l \in S}\left(\tau_{il}(t)\right)^{\delta} * (\eta_{il})^{\varepsilon}}, & Si\ j \in S \\ 0, & Si\ j \notin S \end{cases} \tag{2.4}$$

où δ et ε sont respectivement les exposants liés à l'importance de la trace de phéromones et de la visibilité dans la règle de choix. Les *phéromones* représentent ce qui a été fait dans les cycles passés. Les phéromones sont initialisées à faibles coûts. La mise à jour des phéromones s'effectue après la construction d'une fourmi et toutes les fourmis mettent à jour les traces de phéromones. La mise à jour se calcule à l'aide de l'Équation 2.5.

$$\tau_{ij}(t+1) = (1-\rho) * \tau_{ij} + \Delta\tau_{ij}(t) \tag{2.5}$$

où $\Delta\tau_{ij}(t) = \sum_{k=1}^{K} \tau_{ij}^{k}(t)$. La quantité du dépôt de phéromone d'une fourmi (τ_{ij}^{k}) est directement reliée à la qualité de sa solution, tel que le démontre l'Équation 2.6.

$$\tau_{ij}^{k} = \begin{cases} \frac{X}{f(x)^{k}}, Si\ (i,j) \epsilon\ au\ trajet\ de\ k \\ 0,\ Si\ (i,j) \notin au\ trajet\ de\ k \end{cases} \quad (2.6)$$

où X est un paramètre fixé.

Le dépôt s'accomplit en ajoutant une petite quantité de phéromones sur l'ensemble de son parcours au travers des villes. Dans les premiers cycles, les phéromones forcent les fourmis à explorer l'espace de recherche. Plus les cycles avancent et plus les fourmis sont portées à converger vers la meilleure fourmi. La *visibilité* (η) est liée au problème. Dans le cadre de la règle de choix, les phéromones encouragent la sélection des villes ayant le mieux réussi dans le passé. La visibilité va, pour sa part, encourager la sélection des villes les plus appropriées pour le problème. Pour le VC, la visibilité est régulièrement l'inverse de la distance entre les villes : $\eta_{ij} = \frac{1}{l_{ij}}$.

2.6.1.2. MMAS pour le problème de voyageur de commerce (VC)

Des versions plus évoluées de l'AS sont présentées dans la littérature. Le min-max ant system (MMAS) (Stützle et al. 2000) propose quelques modifications à l'AS. D'abord, seule la meilleure fourmi met à jour les phéromones à la fin d'un cycle. Ensuite, la trace de phéromones est bornée par τ_{min} et τ_{max}. L'initialisation de la trace s'effectue à τ_{max}. La mise à jour des phéromones s'évalue en utilisant l'Équation 2.7.

$$\tau_{ij}(t+1) = \rho * \tau_{ij}(t) + \Delta\tau_{ij} \quad (2.7)$$

où ρ représente le facteur d'évaporation de la trace et $\Delta\tau_{ij} = {}^{1}/_{l^{*}}$ où l^{*} représente la valeur de la fonction objectif de la meilleure fourmi. La différence majeure est dans la détermination de la meilleure fourmi. Avec un MMAS, la meilleure fourmi peut être basée

sur deux possibilités : la meilleure fourmi du cycle « iteration_best » ($fourmi_{ib}$) ou la meilleure fourmi depuis le début « global_best » ($fourmi_{gb}$). La $fourmi_{gb}$ faisant l'exploitation de la meilleure solution trouvée, il y a des risques élevés de se retrouver dans un optimal local. Pour la $fourmi_{ib}$, le risque d'optimal local est moins élevé car elle explore les solutions réalisables. La stratégie proposée par Stützle and Hoos (2000) est d'utiliser les deux fourmis ($fourmi_{gb}\ et\ fourmi_{ib}$) à divers intervalles de cycle. Par exemple, tous les 25 cycles en commençant par la $fourmi_{ib}$ et en alternant avec la $fourmi_{gb}$. La dernière modification de la version AS pour le MMAS est que les traces de phéromones peuvent être réinitialisées.

2.6.1.3. ACS pour le problème de voyageur de commerce (VC)

Le ant colony system (ACS) (Dorigo et al. 1997) est une évolution du AS. Au lieu de la règle de choix comme l'AS, l'ACS propose d'utiliser une règle de transition. Cette règle consiste à choisir la prochaine ville j à insérer dans la fourmi k à la suite de la ville i selon deux fonctionnalités : déterministe ou probabiliste. La fonction *déterministe* a pour effet que la fourmi exploite les solutions réalisables. Quant à la fonction *probabiliste*, elle effectue plutôt de l'exploration des solutions réalisables. Le paramètre seuil (q_0) pondère le choix entre les deux fonctionnalités. Formellement, la règle de transition est définie à l'Équation 2.8.

$$j = \begin{cases} \max_{h \notin S_k}\left([\tau_{ih}]^{\delta} * [\eta_{ih}]^{\varepsilon}\right),\ si\ q \leq q_0 \\ J,\ si\ q > q_0 \end{cases} \qquad (2.8)$$

où q_0 est le seuil qui détermine si la règle de transition est *déterministe* quand $q \leq q_0$, ou *probabiliste* quand $q > q_0$. Dans le cas d'une règle probabiliste, la tâche j est choisie suivant la tâche i selon la probabilité de l'Équation 2.9 :

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\delta} * [\eta_{ij}]^{\varepsilon}}{\sum_{h \notin S_k} [\tau_{ih}]^{\delta} * [\eta_{ih}]^{\varepsilon}} \tag{2.9}$$

où δ et ε représente respectivement les exposants liés à l'importance de la trace de phéromones et de la visibilité. τ_{ij} représente la trace de phéromones entre les villes i et j. Finalement, η_{ij} représente la visibilité de la fourmi entre les villes i et j. Dans le cas du VC, η_{ij} correspond à la distance inversée ($1/l_{ij}$), car la sélection de la ville présentant une petite distance est priorisée.

La mise à jour locale des phéromones est réalisée après la construction d'une fourmi en diminuant la quantité de phéromone des arcs visités. Cette diminution favorise la diversification par la prise en compte des trajets non explorés (Dréo et al. 2003). La mise à jour locale est définie par l'Équation 2.10.

$$\tau_{ij}(t) = \rho * \tau_{ij}(t) + (\rho - 1) * \Delta\tau_{ij} \tag{2.10}$$

où ρ représente la persistance de la trace de phéromones et $\Delta\tau_{ij} = \tau_0$ la trace initiale de phéromones.

La mise à jour globale des phéromones est réalisée en fin de cycle. Seule la meilleure fourmi effectue la mise à jour proportionnellement à la qualité de sa solution. La mise à jour à pour but d'augmenter légèrement la trace de phéromones le long du trajet emprunté par la fourmi, ce qui participe à l'intensification. Elle se définit de la même façon que la mise à jour locale et est présentée à l'Équation 2.11,

$$\tau_{ij}(t) = \rho * \tau_{ij}(t) + (\rho - 1) * \Delta\tau_{ij} \qquad (2.11)$$

à la différence que $\Delta\tau_{ij} = {}^{1}/_{l^*}$ où l^* est la fonction objectif de la meilleure fourmi trouvée jusqu'à maintenant.

Plusieurs adaptations des approches OCF ont été proposées pour traiter l'optimisation multi-objectifs, la prochaine section en fait état.

2.6.2. OCF multi-objectifs Pareto

Gambardella et al. (1999) sont parmi les premiers à proposer une méthode de résolution pour un PMO en utilisant l'AS. La méthode MACS (Multiple Ant Colony System) permet la résolution d'un problème de routage de véhicule avec fenêtre de temps en proposant une approche d'agrégation sur les fonctions objectif. Plusieurs chercheurs ont suivi ce courant en présentant des approches non-Pareto (Doerner et al. 2003; Gravel et al. 2002; Mariano et al. 1999). En 2004, Dorigo et Stützle émettent la constatation que peu de méthodes approximant un ensemble PO à l'aide d'approche Pareto sont disponibles dans la littérature. Cette constatation a été confirmée lors de la revue de la littérature.

Les algorithmes d'OCF utilisant des approches Pareto sont regroupés de diverses façons dans la littérature. Garcia-Martinez et al (2007) amènent une classification selon le nombre de matrices de phéromones et de visibilité. D'autres auteurs se basent sur les composantes de méthode de l'OCF telles que le nombre de matrices de phéromones, la méthode de construction, la méthode est-elle Pareto ou non, la mise à jour et l'évaporation des phéromones ainsi que l'archive des solutions non-dominées (Angus et al. 2009). Dans le cadre de ce mémoire, les approches sont divisées en deux groupes : en multi-colonies et en uni-colonie. Cette division est réalisée dans l'optique de conserver une unité entre la

littérature et le travail proposé. Les approches des deux groupes sont basées sur les approches d'OCF présentées aux Sections 2.6.1.1, 2.6.1.2 et 2.6.1.3, respectivement AS, MMAS et ACS. Le Tableau 2 présente une classification des approches d'OCF Pareto proposées dans la littérature.

Auteurs	Multi-colonies	Uni-colonie	AS	MMAS	ACS
(Iredi et al. 2001)	X	X	X		
(Guntsch et al. 2003)		X		X	
(López-Ibáñez et al. 2004)	X			X	
(Liu et al. 2005)		X	X		
(Angus 2007)		X	X		
(Chaharsooghi et al. 2008)		X	X		
(Yang et al. 2010)		X			X
(Berrichi et al. 2010)	X		X		
(Moncayo-Martinez et al. 2011)	X		X		
(Teixeira et al. 2012)	X			X	
		Multi-colonies (5 articles)	3/5	2/5	0/5
		Uni-colonie (6 articles)	4/6	1/6	1/6

Tableau 2 : Approches d'OCF de base utilisées selon l'article de la revue de la littérature

Les deux dernières lignes du Tableau 2 indiquent le nombre d'articles qui traitent de chaque approche d'OCF. Il est possible de constater que quatre articles sur six (66%) traitent une méthode uni-colonie à l'aide de l'approche AS.

2.6.2.1. Approches multi-colonies

Tel que le démontre le Tableau 2, les approches du groupe *multi-colonies* sont basées à 60% sur un AS et à 40% sur un MMAS. Chaque colonie possède sa phéromone et sa visibilité. Le nombre de colonies est généralement supérieur au nombre d'objectifs. De plus, les méthodes utilisent un nombre de fourmis supérieur aux recommandations formulées par les algorithmes de base de l'AS et du MMAS en uni-objectif. Tel que le démontre la revue de la littérature à la Section 2.5, l'utilisation de concepts des AE Pareto,

le facteur de dominance et le facteur d'isolement, génèrent de bons résultats. Malgré tout, peu d'approches d'OCF de la littérature utilisent les concepts des AE Pareto. Seul Berrichi et al (2010) emploient le tri par front du NSGA-II pour la mise à jour des phéromones.

L'algorithme bi-criterion ant multicolony (BCMC) proposé par Iredi et al (2001) pour un problème bi-objectifs de machine unique minimisant la pénalité de retard et le coût de réglage est traité dans ce mémoire. Le BCMC est présenté ici car il est un exemple d'approche d'OCF n'utilisant pas les facteurs de dominance et d'isolement. L'algorithme du BCMC est présenté à l'aide du pseudo-code de la Figure 5.

1 : Initialiser l'archive locale et globale à $\emptyset$
2 : Initialiser les colonies à $\emptyset$ et leurs matrices de phéromones et de visibilités
3 : $t = 0$
4 : Tant que t < le nombre de cycles maximaux
5 : Pour $col \leftarrow 0$ à $nbColonie$
6 : $\lambda \leftarrow$ calculer la règle d'échange à l'aide des Équations 2.10, 2.11 ou 2.12 selon le cas
7 : Pour $k \leftarrow 0$ à K
8 : Initialiser l'ensemble des tâches admissibles dans S
9 : Pour $i \leftarrow 0$ à taille du problème
10 : Choisir une tâche $j \in S$ avec la probabilité

$$p_{ij}^{k} \leftarrow \frac{\left(\tau_{ij}^{*}\right)^{\lambda\delta} * {\tau'_{ij}}^{(1-\lambda)\varepsilon} * {\eta_{ij}}^{\lambda\delta} * {\eta_{ij}}^{(1-\lambda)\varepsilon}}{\sum_{h\epsilon S}(\tau_{ih}^{*})^{\lambda\delta} * {\tau'_{ih}}^{(1-\lambda)\varepsilon} * {\eta_{ih}}^{\lambda\delta} * {\eta_{ih}}^{(1-\lambda)\varepsilon}}$$

11 : $S \leftarrow S - j$
12 : Évaluer la fourmi k
13 : Mise à jour de l'archive locale et globale
14 : Évaporation des matrices de phéromone de la colonie col
15 : Mise à jour des phéromones
16 : $t++$

Figure 5 : Pseudo-code BiCriterion Ant MultiColony pour un problème bi-objectifs de machine unique minimisant la pénalité de retard et le coût du réglage.(Iredi et al. 2001)

Le BCMC est basé sur l'approche AS avec la particularité d'avoir plusieurs colonies. Le nombre de colonies est, dans le cas du BCMC, supérieur au nombre d'objectifs du

problème. BCMC est de nature multi-colonie et les colonies sont indépendantes les unes des autres. Il est donc essentiel d'avoir un mécanisme qui va permettre aux colonies d'échanger de l'information. Dans le cas contraire, l'ensemble PO généré sera disjoint. La règle d'échange, calculée à la ligne 6 du pseudo-code présenté à la Figure 5, a pour but d'effectuer la jonction entre les objectifs pour s'assurer d'avoir un ensemble PO continu. Ceci se réalise dans le calcul de probabilité de la règle de choix (ligne 10). La règle d'échange est un exposant à chaque composante du calcul qui va permettre de pondérer le choix de la prochaine tâche entre les deux objectifs.

BCMC propose trois mécanismes qui sont appelés règle d'échange (λ). Le premier mécanisme de la règle d'échange est par fourmi (λ_k). Ce mécanisme (λ_k) mesure l'influence relative des objectifs à optimiser (Iredi et al. 2001). Le mécanisme est défini mathématiquement ainsi :

$$\lambda_k = \frac{k-1}{({}^{K}/_{nbColonie})-1} \tag{2.12}$$

où k est la fourmi courante, K est le nombre total de fourmis et $noColonie$ est le nombre total de colonies. Le deuxième mécanisme de la règle d'échange est un intervalle disjoint. Dans ce mécanisme, λ est par colonie (λ_{col}) et représenté ainsi :

$$\lambda_{col} = (col - 1) * (k/nbColonie) + k \tag{2.13}$$

où col représente la colonie courante. Le troisième mécanisme de la règle d'échange permet l'échange d'informations entre les colonies tel que le démontre la Figure 6.

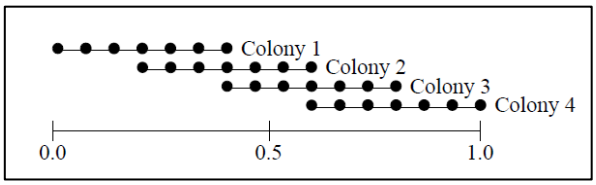


Figure 6 : Mécanisme d'intervalle d'échange à 50% : 4 colonies et 7 fourmis par colonie (Iredi et al. 2001)

Le concept est que la colonie actuelle échange 50% de ses informations avec la colonie précédente. Chaque colonie a un intervalle d'où est tirée aléatoirement la valeur (λ_{Icol}), tel que le présente l'Équation 2.14.

$$\lambda_{Icol} \in \left[(col-1) \Big/ (nbColonie+1), (col+1) \Big/ (nbColonie+1) \right] \qquad (2.14)$$

BCMC utilise une règle de choix présentée à la ligne 10 de la Figure 5. La règle de choix est dirigée par deux phéromones et deux visibilités, soit une phéromone et une visibilité par objectif. Chaque colonie possède deux phéromones et deux visibilités.

Toutes les fourmis de l'ensemble PO effectuent la mise à jour des phéromones. BCMC propose deux moyens de réaliser la mise à jour : par origine ou par région. La Figure 7 illustre la méthode de mise à jour des phéromones par origine pour un BCMC à trois colonies.

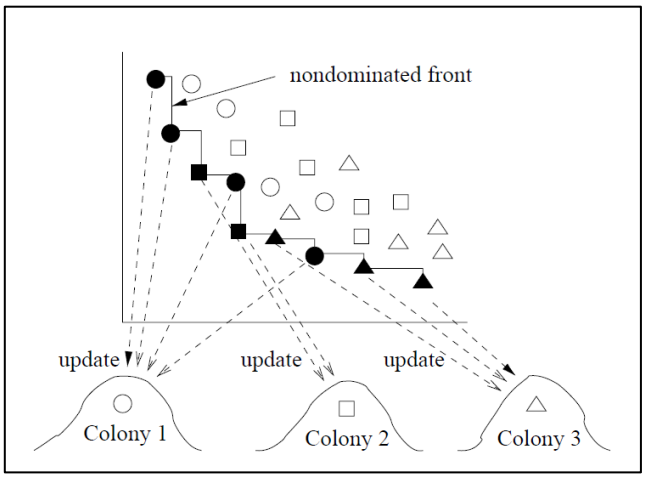


Figure 7 : Mise à jour des phéromones par origine pour trois colonies (Iredi et al. 2001)

L'ensemble PO est représenté par les symboles foncés. Les cercles sont des solutions générées par la colonie 1, les carrées sont des solutions générées par la colonie 2 et ainsi de suite. La méthode par origine considère que les solutions mettent à jour les phéromones de leur colonie d'origine. La Figure 8 illustre la méthode de mise à jour des phéromones par région pour un BCMC à trois colonies.

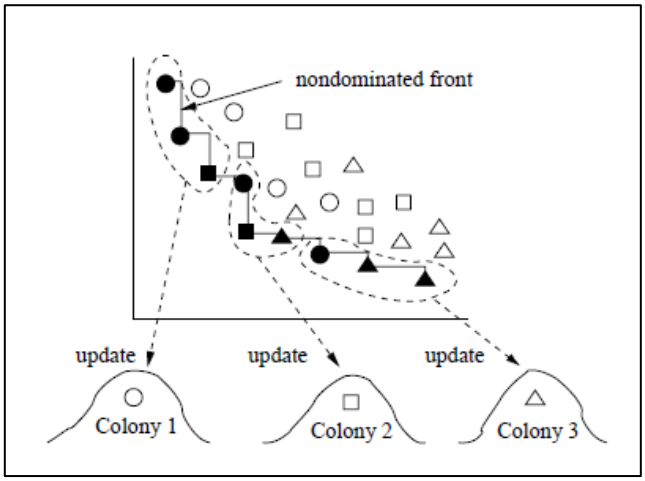


Figure 8 : Mise à jour des phéromones par région pour trois colonies (Iredi et al. 2001)

Tout comme la figure précédente, les symboles foncés représentent l'ensemble PO. Les cercles sont des solutions générées par la colonie 1, les carrés sont des solutions générées par la colonie 2 et ainsi de suite. La méthode par région divise en trois parties l'ensemble PO car il y a trois colonies dans ce cas-ci. La partie i effectue la mise à jour des phéromones de la colonie i où $i = (1,2,3)$.

2.6.2.2. Approches uni-colonie

Tel que le démontre le Tableau 2, les méthodes *uni-colonies* sont basées à 66% sur un AS, à 17% sur un MMAS et à 17% sur un ACS. Ces pourcentages démontrent que la plupart des méthodes uni-colonie utilisent l'approche AS comme base malgré que l'ACS est démontré plus performant que l'AS (Dorigo et al. 1997).

Les approches AS, MMAS et ACS recommandent un nombre de fourmis avoisinant les 20 fourmis. Cette recommandation se justifie par le fait que le nombre de fourmis a un impact sur la performance de l'approche tant au niveau du temps d'exécution que de sa capacité à se diversifier. Ces recommandations ne sont pas suivies dans les méthodes Pareto multi-colonies étudiées car le nombre de fourmis est supérieur aux recommandations faites en uni-objectif.

Les concepts d'AE Pareto, le facteur de dominance et le facteur d'isolement, ont démontré de bons résultats et sont utilisés dans plusieurs méthodes AE Pareto entre autres NSGA-II, PMS^{MO} et GISMOO. Toutefois, peu d'articles utilisant des approches OCF en multi-objectifs uni-colonie utilisent ces concepts. Les seuls dénombrés sont l'article d'Angus (2007) et de Yang et al. (2010). Le premier propose d'utiliser le tri par front du NSGA-II dans le cadre de la mise à jour des phéromones. Dans le même cadre, le second

article propose plutôt d'utiliser une méthode de niching qui se rapproche du facteur d'isolement. Il est possible de remarquer que peu d'articles utilisent les concepts d'AE Pareto et ceux qui les emploient ne font qu'un sur deux : facteur de dominance pour l'article d'Angus et facteur d'isolement pour celui de Yang et al (2010).

Angus (2007) propose également une nouvelle façon de considérer l'exposant lié à l'importance de la visibilité. Chaque fourmi de la colonie possède son exposant au lieu d'être commun à toutes les fourmis. Cette façon de faire permet de diversifier les fourmis dans l'espace de recherche.

Les caractéristiques communes des deux groupes, multi-colonies et uni-colonie, sont à l'effet que les méthodes proposées sont majoritairement basées sur l'approche AS (Tableau 2). De plus, peu de méthodes utilisent les concepts AE Pareto tel que le facteur de dominance et le facteur d'isolement. La Section 2.6.2.1 présente un algorithme représentatif du groupe multi-colonie : l'algorithme BCMC. Étant donné qu'aucun algorithme représentatif du groupe uni-colonie n'a été repéré dans la revue de la littérature, la Section 2.6.2.2 présente les points retenus pour ce groupe, entre autres la contribution d'Angus (2007) dans la considération de l'exposant lié à l'importance de la visibilité.

L'évaluation des méthodes de résolution multi-objectifs s'effectue par l'analyse de leur ensemble PO. La comparaison entre les ensembles PO est plus complexe que celle entre deux solutions uni-objectif. Cette complexité est due à la nature d'ordre partiel entre les solutions des ensemble PO. Également, les ensembles PO doivent être comparés selon les caractéristiques vues à la Section 2.4 : qualité et distribution. La prochaine section présente des métriques pour l'évaluation de la qualité d'ensembles PO.

2.7. Les Métriques

En optimisation multi-objectifs, il est essentiel d'évaluer les ensembles PO avec plusieurs métriques pour s'assurer d'une bonne comparaison entre les ensembles. Les métriques évaluent, habituellement, la qualité des solutions ou la distribution des solutions d'un ou de deux ensembles PO. Afin de proposer une méthodologie valable pour la comparaison des résultats, au moins deux métriques sont nécessaires. D'ailleurs, les solutions d'un ensemble PO ont une relation d'ordre partiel entre elles.

Les métriques ont plusieurs caractéristiques tel que considérer que l'ensemble PO théorique est connu ou inconnu. Une autre caractéristique est que la métrique évalue les ensembles PO individuellement ou deux à deux. Dans les sections suivantes, une liste non-exhaustive de métriques est présentée.

2.7.1. La métrique Rapport d'erreur

La métrique est proposée par Van Veldhuzen (1999). Elle considère que l'ensemble PO théorique est connu. Elle évalue un ensemble PO à la fois par rapport à l'ensemble PO théorique. Cette métrique mesure la non-qualité des solutions, soit dans ce cas-ci la non-convergence d'un ensemble PO vers un ensemble PO théorique. Une valeur de métrique près de un démontre que l'ensemble PO n'a pas convergé vers l'ensemble PO théorique.

2.7.2. La métrique Espacement

La métrique proposée par Schott (1995) considère que l'ensemble PO théorique est inconnu. Elle mesure la distribution des solutions d'un ensemble PO à la fois. Plus précisément, la métrique mesure l'uniformité de la répartition des solutions de l'ensemble PO (Collette et al. 2002). Formellement, la métrique se définit ainsi :

$$espacement = \left[\frac{1}{1-o} * \sum_{i=1}^{n} (\bar{g} - g_i)^2\right]^{1/2} \quad (2.15)$$

où $g_i = \min_j \left(\left|f_1^i(x) - f_1^j(x)\right| + \left|f_2^i(x) - f_2^j(x)\right|\right)$; $i, j = 1, \ldots, o$; $i \neq j$; $\bar{g}$ est la moyenne de toutes les variables g_i et o est le nombre d'éléments dans l'ensemble PO.

2.7.3. La métrique Hole Relative Size (HRS)

La métrique Espacement, présentée à la Section 2.7.2, a tendance à dissimuler les écarts importants (Collette et al. 2002). Collette et Siarry (2002) proposent la métrique HRS qui mesure la taille du plus grand interstice dans l'ensemble PO. Les caractéristiques demeurent les mêmes que la métrique Espacement. Elle se définit mathématiquement ainsi :

$$HRS = \frac{\max_i g_i}{\bar{g}} \quad (2.16)$$

où $g_i = \min_j \left(\left|f_1^i(x) - f_1^j(x)\right| + \left|f_2^i(x) - f_2^j(x)\right|\right)$; $i = 1, \ldots, o$ et $\bar{g}$ est la moyenne de toutes les variables g_i.

2.7.4. La métrique Hypervolume (H)

La métrique stochastique proposée par Zitzler (1999) considère que l'ensemble PO théorique est inconnu. Elle mesure la qualité des solutions d'un ensemble PO en estimant le volume situé sous la courbe d'un ensemble PO. Dans le cas d'un problème à deux objectifs, ce calcul se résume en une aire. Plus formellement, H calcule le volume inclus entre l'union des polytopes $\varphi_1, \varphi, \dots, \varphi_n$ et passant par les points $f_1(x), f_2(x), \dots, f_n(x)$ où φ_i est formé par l'intersection des hyperplans résultants de x_i tel que le démontre la Figure 9. De plus, φ_i est représenté par un rectangle délimité par les points $(0,0)$ et $\big(f_1(x_1), f_2(x_2)\big)$.

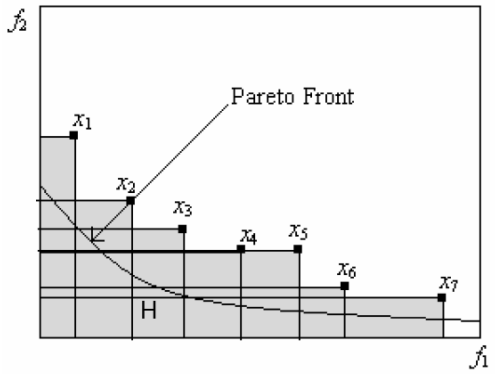


Figure 9 : Métrique H pour un cas à deux objectifs et 7 variables $(x_1, x_2, \dots x_7)$ en minimisation (Grosan 2003)

2.7.5. La métrique Couverture $(couv)$

La Couverture $(couv)$ (Zitzler 1999) est une métrique qui considère que l'ensemble PO théorique est inconnu. Elle mesure la qualité des solutions en comparant des surfaces de compromis entre deux méthodes. Soit A et B, deux ensembles de solutions non-dominées. $couv(A, B)$ donne le pourcentage d'éléments de B dominés par au moins un élément de A à l'aide de cette formule :

$$couv(A,B) = \frac{|\{b \in B | \exists a \in A : a \succcurlyeq b\}|}{|B|} \quad (2.17)$$

où $\succcurlyeq$ est le symbole qui détermine une dominance faible; donc $a \succcurlyeq b$ signifie que a n'est pas pire que b sur tous les objectifs et $|\ \ |$ représente la cardinalité. Il est à noter que si $couv(A,B) = 1$ alors tous les éléments de B sont dominées par ceux de A. Advenant que $couv(A,B) = 0$, alors aucun élément de B n'est dominé par ceux de A. De plus, $couv(A,B)$ n'est pas nécessairement l'inverse de $1 - couv(B,A)$.

2.7.6. La métrique Différence (D)

La métrique proposée par Zitzler (1999) considère que l'ensemble PO théorique est inconnu. Elle mesure la qualité des solutions en comparant deux à deux les ensembles PO. Cette métrique palie au désavantage de la métrique $couv$. La métrique $couv$ peut calculer des résultats erronés si les deux ensembles PO se croisent. À l'aide de la métrique H, D propose une différence de couverture entre deux ensembles PO. Elle est formellement décrite ainsi : $D(A,B) = H(A+B) - H(B)$ où A et B sont deux ensembles PO et $H(A)$ est l'Hypervolume (H) de A.

2.7.7. La métrique de Laumanns

La métrique proposée par Laumanns et al. (2000) considère que l'ensemble PO théorique est inconnu. Elle mesure la qualité des solutions sur un ensemble PO. Cette métrique est basée sur la mesure de Lebesque. En général, elle se rapproche considérablement de la métrique H de Zitzler présentée à la Section 2.7.4 à la différence qu'elle calcule une valeur entre 0 et 1.

2.7.8. La métrique d'Espacement Δ

La métrique proposée par Deb et al. (2000) considère que l'ensemble PO théorique est connu. Elle mesure la distribution des solutions sur un ensemble PO. Essentiellement, la métrique calcule une distance euclidienne entre les solutions et normalise le résultat avec la cardinalité de l'ensemble PO théorique. Le principal défaut de cette métrique est qu'elle n'est adaptée que pour les problèmes à deux objectifs.

La dernière section a présenté trois métriques analysant la distribution et cinq métriques analysant la qualité des solutions d'ensembles PO. Les métriques choisies dans le cadre de l'analyse des résultats de ce mémoire sont l'Hypervolume, la Couverture et l'Espacement.

2.8. Objectifs et méthodologie de la recherche

La revue de la littérature a permis de faire ressortir plusieurs observations qui permettent de poser les différents objectifs. La première observation est que beaucoup d'entreprises de différents secteurs d'activité sont confrontées à des problèmes d'ordonnancement multi-objectifs. Les problèmes traités en théorie contiennent généralement des aspects se rapprochant des contextes réels. Toutefois, la simplification de certains aspects tels que la considération uni-objectif des problèmes éloigne la théorie de la pratique. Arroyo et al (2011) proposent un problème multi-objectifs (MURMO) qui considère plusieurs aspects des contextes réels dont la considération multi-objectifs. La seconde observation est qu'à notre connaissance, une seule méthode résout MURMO : le MOVNS3 (Arroyo et al. 2011), qui est une méthode de recherche dans la structure de voisinage utilisant l'approche Pareto. L'observation suivante concerne les approches

Pareto, qui représentent une des façons de traiter la fonction objectif pour résoudre un PMO. Cette approche utilise la notion dominance qui est discutée à la Section 2.4. Pour finir les observations, plusieurs méthodes ont été adaptées pour solutionner des PMO avec une approche Pareto. Les AE connaissent de bons résultats avec NSGA-II (Deb et al. 2000), PMS^{MO}(Zinflou et al. 2008), GISMOO (Zinflou et al. 2012), SPEA (Zitzler et al. 2001) et PESA-II (Corne et al. 2001). Également, plusieurs méthodes de la famille d'OCF Pareto sont présentées au Tableau 2. Parmi les méthodes multi-colonies, on rettrouve le *Ant Colony Optimization for bi-objective Quadratic Assignment Problem* (ACO-bQAP) (López-Ibáñez et al. 2004) et le *Multi-Objective Ant Colony Optimization* (MOACO) (Berrichi et al. 2010). En ce qui concerne les méthodes uni-colonie, le *Multi-Objective-ant* (MO-ant) (Liu et al. 2005) et le *Crowding Population-based Ant colony Optimization* (CPACO) (Angus 2007) sont par exemple identifiées.

L'objectif principal de ce mémoire est de *proposer des approches de résolutions multi-objectives performantes afin de permettre un rapprochement entre la théorie et la pratique.* La revue de la littérature a démontré qu'un écart entre la théorie et la pratique existe. Le problème MURMO, utilisé comme problème de support dans ce mémoire, permet de traiter certains aspects représentatifs de contexte réel.

La méthodologie utilisée pour atteindre cet objectif est de présenter des méthodes Pareto pour résoudre le problème de support MURMO. Étant donné qu'une seule méthode a été proposée jusqu'à maintenant pour le problème MURMO, le MOVNS3, ce mémoire propose donc l'adaptation de trois algorithmes de la littérature pour le problème afin d'avoir un ensemble de résultats de référence : NSGA-II (Deb et al. 2000), PMS^{MO} (Zinflou

et al. 2008) et GISMOO (Zinflou et al. 2012). Ces adaptations ont fait l'objet d'un acte de conférence avec comité de lecture pour la conférence ROADEF 2013 (Gagné et al. 2013).

L'objectif principal engendre plusieurs objectifs secondaires qui seront présentés dans les prochains paragraphes. Le premier est de *résoudre efficacement un PMO ayant des aspects traitant de contextes réels.* Les solutions d'une méthode de résolution Pareto doivent converger vers le front Pareto et être bien distribuées le long de la frontière. L'objectif suggère aussi que le PMO doit avoir des aspects se rapprochant des contextes réels. Le PMO choisi est le problème MURMO.

Pour évaluer l'efficacité d'une méthode, l'ensemble des solutions non-dominées est analysé. Cette analyse s'effectue à l'aide de différentes métriques. Dans le cadre de ce mémoire, les métriques employées qui évaluent la qualité des solutions sont : l'Hypervolume (Zitzler 1999) et la Couverture (Zitzler 1999). Pour l'évaluation de la distribution des solutions, la métrique est l'Espacement (Schott 1995).

Le deuxième objectif secondaire consiste à proposer *une comparaison équitable entre une méthode multi-colonies et une méthode uni-colonie.* La revue de la littérature du mémoire présente un regroupement des méthodes multi-objectifs de la famille d'OCF en deux groupes : les méthodes multi-colonies et les méthodes uni-colonie. Les conclusions du regroupement sont que la majorité des approches d'OCF sont basées sur l'AS, la version de base des algorithmes dans ce domaine, et que peu de méthodes utilisent des concepts AE Pareto.

La méthode multi-colonies choisie est le bi-criterion ant multi-colony (BCMC) de Iredi et al (2001). Le BCMC est choisi car c'est une méthode multi-objectifs n'utilisant pas

de concepts empruntés aux AE Pareto. Pour représenter les groupes des méthodes uni-colonie, l'approche d'OCF ant colony immune system for multiple objective optimization (ACISMOO), qui est une transposition de GISMOO adapté au problème MURMO, est utilisée. GISMOO a été choisi, car il s'agit de la méthode qui s'est avérée la meilleure parmi NSGA-II (Deb et al. 2000), PMS^{MO} (Zinflou et al. 2008) et GISMOO (Zinflou et al. 2012). Une comparaison équitable est proposée entre BCMC et ACISMOO pour le problème MURMO. Il est possible de qualifier la comparaison d'équitable car les conditions d'expérimentation sont équivalentes pour les deux méthodes (Section 4.1). Outre les conditions d'expérimentation, l'ensemble des méthodes comparées dans ce mémoire sont programmées à l'aide de MetLib. Cet outil propriétaire permet l'accès à plusieurs algorithmes tels que NSGA-II, PMS^{MO} et GISMOO. Aussi, cette bibliothèque comprend plusieurs outils de gestion des solutions non-dominées. Un tel outil permet d'offrir un même environnement de programmation aux méthodes.

Le troisième objectif secondaire présente *la transposition de concepts AE Pareto dans une approche d'OCF.* La revue de la littérature démontre que peu d'approches d'OCF Pareto utilisent des concepts AE Pareto. Le tri par front du NSGA-II est utilisé dans les articles de Berrichi (2010) et Angus (2007). De plus, une méthode de niching comparable à un facteur d'isolement est employée dans l'article de Yang et al (2010). Toutefois, les concepts AE Pareto tels que le facteur de dominance et d'isolement améliorent les méthodes AE (Deb et al. 2000; Zinflou et al. 2012; Zinflou et al. 2008).

C’est par la transposition de GISMOO vers un ACS que cet objectif est répondu. Cette transposition permet l’utilisation de concepts AE Pareto tels que le facteur de dominance et le facteur d’isolement au sein de la méthode.

Le dernier objectif consiste à *suggérer une bonification de la méthode ACISMOO.* Les méthodes uni-colonie doivent permettre à chacune de leurs fourmis de converger vers le front Pareto ou de se diversifier. L’amélioration propose de faire varier les composantes de la fourmi de telle sorte qu’une fourmi peut converger ou se diversifier en cours d’exécution de la méthode. L’amélioration d’ACISMOO est comparée avec la version originale de l’algorithme. La performance des algorithmes est évaluée à l’aide des métriques.

La première amélioration d’ACISMOO est l’ajout de différentes visibilités. La seconde amélioration consiste à ce que chaque fourmi de la colonie calcule la visibilité de façon différente, donc si deux fourmis ont le même déterminant, elles ne sélectionnent pas nécessairement la même tâche. Ces améliorations apportent de la diversité à la version originale d’ACISMOO.

CHAPITRE 3

COMPARAISON MULTI/UNI-COLONIES POUR UN PROBLÈME DE MACHINE UNIQUE AVEC CARACTÉRISTIQUES SE RAPPROCHANT DE CONTEXTES RÉELS

3.1 Introduction

La machine unique est une configuration classique dans la littérature en ordonnancement. En industrie, les problèmes d'ordonnancement sont généralement plus complexes qu'une machine unique. Toutefois, il est possible de les décomposer en problèmes plus simples qui se modélisent en une machine unique. Par exemple, dans le problème de goulot d'étranglement dans un système de production complexe, le goulot peut être vu comme une machine unique. En optimisant le goulot, tout le système est optimisé (Pinedo 2012). D'ailleurs, la majorité des installations d'entreprises de service peuvent être modélisées avec une machine unique : coiffeuse, esthéticienne, garagiste, pour ne nommer que ceux-ci. L'intérêt académique est tout aussi grand. Pour une configuration de machine unique où chaque tâche a un temps d'exécution et une date de remise, l'objectif de minimisation du retard total est prouvé NP-difficile par Du et Leung (1990). Également avec la même configuration de machine unique, Wan et Yuan (2013) démontrent que la pénalité totale d'avance et retard est NP-difficile au sens fort. Des problèmes ayant une telle complexité créent une grande curiosité académique. Dans le but de se rapprocher des contextes réels, plusieurs extensions de la machine unique ont été proposées (Biskup et al. 2001; Graves et al. 1999; Hall et al. 1991; Liu et al. 2002; Tan et al. 1997). L'extension de la machine unique étudiée dans ce mémoire est celle proposée par Arroyo et al. (2011) intitulée MURMO et est présentée à la Section 3.2. Par la suite, une revue de la littéraire des méthodes traitant de ce problème est présentée à la Section 3.3. Le chapitre se poursuit avec une comparaison entre une méthode multi-colonies et une méthode uni-colonie pour la résolution du problème MURMO. Ces deux méthodes sont respectivement le bi-criterion ant multicolony (BCMC) à la Section 3.4.1

et le ant colony immune system for multiple objective optimization (ACISMOO) à la Section 3.4.2. Enfin, la Section 3.5 présente les améliorations proposées pour l'algorithme ACISMOO.

3.2 Définition du problème MURMO

Causé par des hypothèses simplificatrices émises en théorie de l'ordonnancement, l'écart entre les contextes pratique et académique est régulièrement présent. Toutefois, l'écart se réduit considérablement ces dernières années (Allahverdi et al. 2008). Le problème MURMO tend à réduire cet écart en proposant plusieurs caractéristiques associées aux contextes réels.

La première caractéristique est le réglage dépendant de la séquence. Son inverse, soit le réglage indépendant de la séquence, considère le temps de réglage comme négligeable ou inclus dans le temps d'exécution. Cependant, cette simplification n'est pas représentative de la plupart des contextes réels. Alors, il est nécessaire de considérer les temps de réglage comme étant distincts des temps d'exécution. Cette configuration du réglage est répandue en industrie car 70% des activités industrielles ont des temps de réglage dépendant de la séquence par rapport à 13% où toutes les activités ont des temps de réglage (Conner 2009; Panwalkar et al. 1973). D'ailleurs, plus de deux cents articles traitant de réglage dépendant de la séquence ont été dénombrés par Zhy et Wilhelm (2006).

La deuxième caractéristique est la présence d'une fenêtre d'échéance pour chaque tâche. Ces fenêtres génèrent des pénalités d'avance ou de retard, si la tâche ne se termine pas dans cette fenêtre d’échéance. Dans la littérature, il est courant de traiter les pénalités d'avance ou de retard avec une date de remise commune pour l'ensemble des tâches

(Gordon et al. 2002). Que l'on soit dans le cas d'une date commune ou de fenêtres d'échéance, le traitement des pénalités d'avance et de retard est fréquent en industrie ayant un environnement de juste-à-temps. Ainsi, dans un tel environnement, une tâche doit se terminer le plus près possible de sa date de remise. Dans le cas qui occupe ce mémoire, la date de remise est représentée par une fenêtre d'échéance bornée par une date due au plus tôt et une date due au plus tard. Alors, la tâche doit se terminer aussi près que possible de la date due au plus tôt sans dépasser la date due au plus tard. Un temps de terminaison en avance occasionne des coûts supplémentaires de possessions de marchandises. Ces coûts sont traduits en théorie par une pénalité émise à la tâche ayant un temps de terminaison en avance. À l'inverse, un temps de terminaison en retard occasionne des pertes de clients et/ou de réputation. En théorie, cette perte est également traduite par une pénalité de retard émise à la tâche.

La troisième caractéristique associée aux contextes réels du problème MURMO est la possibilité d'insérer des temps morts dans l'ordonnancement de la machine unique. Les temps morts ne sont habituellement pas permis car ils augmentent le temps total passé dans le système des produits sans toutefois permettre d'en produire davantage. D'ailleurs, de tels temps morts portent les dernières tâches de l'ordonnancement à avoir des pénalités de retard. Néanmoins, pour le problème MURMO, les temps morts sont plus qu'utiles. Ils permettent d'ordonnancer les tâches dans leur fenêtre d'échéance et ainsi éviter les pénalités d'avance et de retard. Tel que l'illustre la Figure 10, la tâche p_j n'aurait pas pu être insérée dans sa fenêtre d'échéance sans avoir préalablement inséré un temps mort.

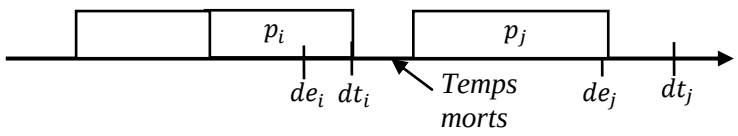


Figure 10 : Illustration d'un temps mort dans un ordonnancement

En pratique, ces temps morts peuvent être utilisés pour la mise au point de la machine. Il est à noter que certains environnements industriels ne peuvent pas avoir de temps morts dans l'ordonnancement de leur machine, alors une forte pénalité peut être appliquée. Toutefois, cette configuration industrielle ne fait pas partie de ce mémoire.

La quatrième caractéristique est la nature multi-objectifs du problème MURMO. Arroyo et al. (2011) mentionnent que la majorité des travaux en machine unique sont uni-objectif. Toutefois, les décideurs sont appelés à résoudre des problèmes ayant plusieurs objectifs contradictoires et à optimiser simultanément. Dans l'ordonnancement de la machine unique, différentes natures d'objectifs peuvent être considérés. Les objectifs les plus fréquemment traités sont la minimisation du temps total de terminaison (makespan), la minimisation du temps total passé dans le système (flowtime) et la minimisation du retard total (total tardiness). Ces objectifs sont justifiés en pratique. Le makespan est lié à la maximisation de l'utilisation des ressources. Pour le flowtime, il est plutôt lié aux temps de production des produits. Aussi, le total tardiness est lié aux dates de remise. Il est à noter que l'objectif total tardiness est le plus important en ordonnancement (Wisner et al. 1995). Le problème visé par ce mémoire traite de la minimisation du flowtime et de la minimisation des pénalités d'avance et de retard.

Suite à la présentation des caractéristiques du problème MURMO se rapprochant des contextes réels, une définition formelle du problème MURMO est énoncée (Arroyo et al. 2011). Supposons m tâches à fabriquer sur une machine unique disponible en

continue. Chaque tâche j est prête pour l'exécution au temps zéro. Une tâche j est définie en fonction de son temps d'exécution p_j, de sa fenêtre d'échéance $[de_j, dt_j]$, où de_j représente la date due au plus tôt et dt_j, celle au plus tard ainsi que des pénalités d'avance α_j et de retard β_j. Pour représenter le fait qu'une tâche qui a un temps de terminaison en retard provoque plus de conséquences négatives qu'une tâche avec un temps de terminaison en avance, dans les instances proposées par Arroyo (2011), la pénalité de retard (β) est toujours plus élevée que la pénalité d'avance (α). Entre deux tâches i et j, un temps de réglage dépendent de la séquence s_{ij} est considéré. Dans ce problème, une tâche doit préférablement se terminer dans sa fenêtre d'échéance $[de_j, dt_j]$. Si le temps de terminaison C_j de la tâche j est dans la fenêtre d'échéance, dit autrement si $C_j \in [de_j, dt_j]$, alors la tâche j n'entraine pas de pénalité d'avance $(E_j = 0)$ ou de retard $(T_j = 0)$. Sinon, une pénalité de d'avance $(\alpha_j E_j)$ ou de retard $(\beta_j T_j)$ est encourue. L'avance et le retard sont respectivement calculés $E_j = \max\{0, de_j - C_j\}$ et $T_j = \max\{0, C_j - dt_j\}$. La résolution de ce problème consiste à trouver une séquence qui minimise la somme pondérée de la pénalité d'avance et de retard (f_1) (Équation 3.1) ainsi que le temps total passé dans le système (f_2) (Équation 3.2). Pour une séquence de tâches $x = (x_1, \dots x_j, \dots, x_m)$ les fonctions objectif f_1 et f_2 sont évaluées ainsi :

$$f_1(x) = \sum_{j=1}^{m} (\alpha_j E_j + \beta_j T_j) \tag{3.1}$$

$$f_2(x) = \sum_{j=1}^{m} C_j \tag{3.2}$$

Tel qu'annoncé plus haut, f_1 entraine des pénalités d'avance et de retard. Cette caractéristique traduit l'incertitude et la tolérance des dates de remise en contexte réel. Dans le but d'ordonnancer les tâches dans leur fenêtre d'échéance, le problème accepte

les temps morts dans la séquence, tel qu'expliqué précédemment. Le calcul des fonctions objectif incluant la possibilité de temps morts devient alors difficile (Wan et al. 2002). Avec la présentation du problème MURMO, Arroyo et al. (2011) proposent d'utiliser la méthode « the optimal timing algorithm » (OTA) (Wan et al. 2002) pour déterminer le temps de terminaison des tâches d'une séquence d'ordonnancement donnée. Cette même méthode est utilisée dans le cadre de ce mémoire dans un but d'égalité de comparaison avec les travaux proposés par Arroyo et al (2011). Étant donné que les fonctions objectif (Équations 3.1 et 3.2) demandent, dans leur calcul, d'avoir les dates de fin (C_j) de chacune des tâches d'une séquence donnée, la méthode OTA est réalisée au préalable pour déterminer ces dates de fin. Par définition, OTA est une méthode gloutonne qui utilise une séquence de tâche fournie par la méthode de résolution pour déterminer les dates de fin des tâches en insérant des temps morts et en ayant pour but d'avoir le plus de tâches se terminant dans leur fenêtre d'échéance. Les dates de fin des tâches sont attribuées dans l'ordre d'apparition des tâches dans la séquence donnée. OTA introduit également le concept de *bloc*. Un *bloc* est un ensemble de tâches n'ayant aucun temps mort entre elles. Les temps morts s'insèrent, alors, entre deux *blocs*. (Wan et al. 2002). Considérons par exemple un problème simplifié à cinq tâches décrit au Tableau 3.

Paramètres	Tâche 1	Tâche 2	Tâche 3	Tâche 4	Tâche 5
Temps d'exécution ($p_{j)}$	9	15	8	12	5
Date due au plus tôt (de_j)	15	150	22	140	21
Date due au plus tard (dt_j)	25	170	30	180	22
Pénalité d'avance (α_j)	3	2	4	1	5
Pénalité de retard (β_j)	7	6	8	4	10

Tableau 3 : Instance à 5 tâches sans temps de réglage (Arroyo et al. 2011)

La séquence de tâches $(1, 5, 3, 4, 2)$ donne un ordonnancement avec OTA tel que décrit à la Figure 11.

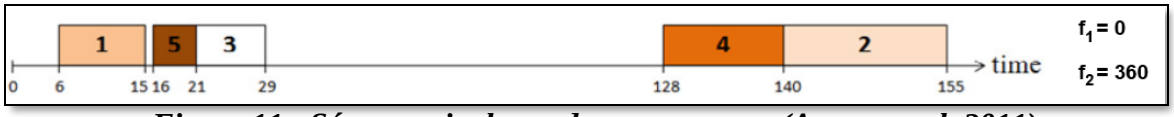


Figure 11 : Séquence incluant des temps morts (Arroyo et al. 2011)

Cette séquence n'encourt aucune pénalité d'avance ou retard, car l'ensemble des tâches est ordonnancé dans leur fenêtre d'échéance. Pour ce faire, OTA a inséré des temps morts aux endroits judicieux tel qu'expliqué précédemment.

À la différence, la séquence de tâches $(5, 4, 1, 2, 3)$ donne un ordonnancement avec OTA décrit à la Figure 12 qui entraine des pénalités d'avance et de retard.

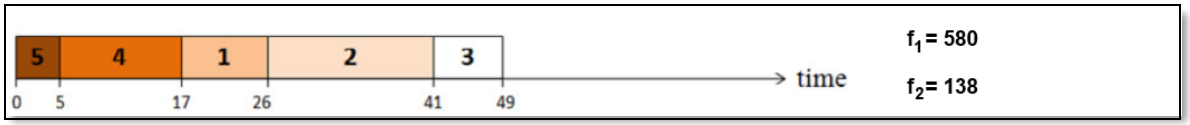


Figure 12 : Séquence sans temps mort (Arroyo et al. 2011)

Ordonnancer les tâches de cette séquence dans leur fenêtre d'échéance est impossible étant donné l'ordre d'apparition des tâches.

3.3 Méthode de résolution du problème MURMO proposée dans la littérature

À la lumière de la revue de la littérature, très peu d'articles traitent de ce problème. Avec la présentation du problème MURMO, Arroyo et al.(2011) proposent le « multi-objective variable neighborhood search 3 » (MOVNS3). Cette méthode est une recherche dans le voisinage incluant un processus d'intensification basé sur la dominance au sens Pareto ainsi que deux structures de voisinage adaptées au problème. Étant donné le peu de recherches réalisées sur le problème MURMO, ce mémoire a généré un ensemble de résultats pouvant servir pour une comparaison future. Pour ce faire, ce mémoire propose l'adaptation de trois méthodes de la littérature pour le problème MURMO, soit NSGA-II (Deb et al. 2000), PMS^{MO} (Zinflou et al. 2008) et GISMOO (Zinflou et al. 2012). Ces trois méthodes de la littérature sont comparées avec MOVNS3 sur les instances utilisées par Arroyo et al (2011). Les conditions d'expérimentation sont détaillées à la Section 4.1. Cette comparaison a fait l'objet d'un acte de conférence lors de la conférence ROADEF 2013 à Troyes.

3.4 Comparaison multi/uni-colonies

Tel qu'expliqué dans les sections précédentes, le problème traité (MURMO) contient plusieurs caractéristiques associées à des contextes réels. À notre connaissance, il y a une seule méthode qui résout le problème MURMO, le MOVNS3. Le mémoire propose l'adaptation de trois algorithmes de la littérature réputés performants : NSGA-II, PMS^{MO} et GISMOO. L'étape suivante bonifie les méthodes de résolution du problème MURMO en proposant des méthodes de résolution constructives.

Dans l'optique d'enrichir la littérature, ce mémoire considère les méthodes de résolution dites constructives : les approches OCF. Lors de la création d'une méthode de

résolution avec les approches OCF en multi-objectifs, plusieurs décisions doivent être prises : quelles approches d'OCF utiliser, combien de colonies ou quels concepts d'AE utilisés? Cette section présente deux méthodes de résolution à l'aide d'approche OCF : une première multi-colonies et une seconde uni-colonie. La comparaison de ces deux méthodes va permettre de connaître l'impact d'utiliser une ou plusieurs colonies dans la résolution du problème bi-objectifs MURMO. De plus, il sera testé si l'adaptation de concepts AE Pareto à des algorithmes d'OCF, tel que les facteurs de dominance et d'isolement, permet d'améliorer les résultats. Les résultats de ceux-ci sont présentés à la Section 4.2. Les Sections 3.4.1 et 3.4.2 présentent respectivement les approches multi-colonies et uni-colonie utilisées.

3.4.1 Méthode multi-colonies : bi-criterion ant multi-colony (BCMC) pour la résolution du problème MURMO

Cette approche a été proposé par Iredi et al (2001). Cette méthode a été choisie car elle n'utilise aucun concept d'AE Pareto. De plus, elle est basée sur l'approche d'OCF AS et elle est multi-colonies. Pour finir, le problème traité par les auteurs se rapproche du problème MURMO, donc permet une adaptation plus fidèle de la méthode. Cette section est réservée à l'adaptation du BCMC pour la résolution du problème MURMO.

BCMC est un algorithme multi-colonies, alors une règle d'échange entre les colonies est obligatoire pour s'assurer d'avoir un ensemble PO continu. La règle utilisée pour l'adaptation du problème MURMO est celle de l'Équation 2.14 présentée à la Section 2.6.2.1. Cette règle est choisie car elle est celle qui trouve un plus grand nombre de solutions dans le centre de l'ensemble PO sur de petits nombres de colonies (2 ou 5).

Dès que le nombre de colonies dépasse dix, la différence entre cette règle et celle de l'Équation 2.13 est minime (Iredi et al. 2001).

BCMC a pour base l'approche d'OCF AS, alors il y a présence d'une règle de choix dont la probabilité de sélectionner la tâche j après la tâche i dans la fourmi k est décrite à l'Équation 3.3.

$$p_{ij}^{k} = \frac{\left(\tau_{ij}^{p}\right)^{\lambda\delta} * \left(\tau_{ij}^{f}\right)^{(1-\lambda)\delta} * \left(\eta_{ij}^{p}\right)^{\lambda\varepsilon} * \left(\eta_{ij}^{f}\right)^{(1-\lambda)\varepsilon}}{\sum_{h \in S}\left(\tau_{ih}^{p}\right)^{\lambda\delta} * \left(\tau_{ih}^{f}\right)^{(1-\lambda)\delta} * \left(\eta_{ih}^{p}\right)^{\lambda\varepsilon} * \left(\eta_{ih}^{f}\right)^{(1-\lambda)\varepsilon}} \tag{3.3}$$

où δ et ε représentent respectivement l'exposant lié à l'importance de la trace de phéromones et de la visibilité, τ^{p} et η^{p} sont respectivement la trace de phéromones et la visibilité pour l'objectif f_1 ainsi que τ^{f} et η^{f} sont la trace de phéromones et la visibilité pour l'objectif f_2. La trace de phéromones de l'objectif f_1 est calculée tel que défini par Iredi et al. Les auteurs suggèrent d'utiliser le calcul de la trace de phéromones proposé par Merkle et Middendorf (2000). Ce calcul prend en considération les décisions passées en introduisant une sommation des traces de phéromones des chemins passés.

La mise à jour des phéromones est globale comme il est courant de le voir en AS. La méthode retenue pour l'adaptation est la mise à jour par région tel que décrit à la Section 2.6.2.1. Cette décision a pour justification que la mise à jour par région donne de meilleurs résultats que la mise à jour par origine, même si les différences sont très légères (Iredi et al. 2001).

La visibilité pour l'adaptation au problème MURMO est différente de ce qui est proposé par Iredi et al., car elle est liée au problème traité. La visibilité $(\eta_{ij}^{f})^{\theta}$ associée au temps total passé dans le système (flowtime (f_2)). Elle priorise les petits temps d'exécution, tel que le démontre l'Équation 3.4.

$$1/p_j + s_{ij} \tag{3.4}$$

La visibilité $\left(\left(\eta_{ij}^{p}\right)^{\varpi}\right)$ visant à optimiser l'objectif de pénalité d'avance et de retard (pénalité (f_1)). Un *déterminant* est préalablement établi afin d'identifier si les tâches restant à ordonnancer dans la séquence sont majoritairement en retard, à temps ou en avance. Avant tout, le déterminant doit être initialisé. Le déterminant est définit en fonction des tâches qui restent à insérer dans la séquence d'une fourmi donnée. Par exemple, le Tableau 4 présente une séquence de cinq tâches dont les tâches 4 et 2 sont déjà ordonnancées.

0	1	2	3	4
4	2	?	?	?

Tableau 4 : séquence de 5 tâches dont les tâches 4 et 2 sont insérées

La tâche qui sera insérée à la position 2 aura un temps de terminaison à situer avant sa fenêtre d'échéance (en avance), après sa fenêtre d'échéance (en retard) ou dans sa fenêtre d'échéance (à temps). En déterminant pour chaque tâche son temps de terminaison si elle est ajoutée à la position courante, il est possible d'établir le Tableau 5 suivant :

En avance	**En retard**	**À temps**	**Déterminant**
∞	0	0	Avance
∞	∞	0	Retard
0	∞	0	
∞	0	∞	À temps
0	0	∞	
0	∞	∞	Plus grand entre en retard et à temps
∞	∞	∞	

Tableau 5 : Initialisation du déterminant

où « 0 » signifie que parmi les tâches possibles à insérer à la position actuelle, aucune n'a un temps de terminaison du nom de la colonne et ∞ signifie qu'une ou plusieurs tâches à insérer à la position actuelle a un temps de terminaison du nom de la colonne. Le Tableau

5 se lit comme suit : sur la première ligne, s'il y a plusieurs tâches en avance, aucune tâche en retard et aucune tâche à temps pour la position actuelle, alors le déterminant est $avance$. Les déterminants *retard* et *à temps* ne considèrent pas les tâches en avance. Ceci se justifie par le fait que si une tâche en avance est ordonnancée avant une tâche qui est déjà en retard, cela a pour conséquence de retarder d'avantage la tâche en retard et ainsi augmenter la pénalité de retard. De plus, ordonnancer une tâche en avance en premier ne fait que retarder la tâche à temps, ce qui risque d'encourir une pénalité de retard. Dans les deux cas précédents, la tâche en avance génère une pénalité d'avance. Advenant qu'il y a plusieurs tâches en retard et plusieurs tâches à temps, le déterminant est initialisé selon le plus grand nombre de tâches. Dit autrement, si pour la position courante, trois tâches sont en retard et deux tâches sont à temps, alors le déterminant est $retard$.

Finalement, il est possible de remarquer à l'Équation 3.3 que l'exposant lié à l'importance de la trace de phéromones est à un. Une telle valeur traduit que la trace est toujours considérée dans la règle de choix.

3.4.2 Méthode uni-colonie : ant colony immune system for multiple objectif optimization (ACISMOO) pour la résolution du problème MURMO

ACISMOO est une transposition de la méthode GSIMOO, présentée à la Section 2.5.3, vers un ACS. Pour permettre une comparaison équitable entre le BCMC et l'ACISMOO, la visibilité et l'évaluation de la fourmi (OTA) demeurent les mêmes. La transposition de GISMOO vers un ACS a été choisie car GISMOO est une méthode évolutionnaire multi-objectifs qui connaît les meilleurs résultats lors de la comparaison entre MOVNS3, NSGA-II ET PMS^{MO}. Cette comparaison est présentée à la Section

4.2.1. Également, cette transposition permet d'avoir une approche OCF qui utilise les facteurs de dominance et d'isolement dans son processus.

La transposition de GISMOO vers un ACS se réalise en remplaçant la partie génétique par un algorithme d'ACS Pareto. Dit autrement, 50% de la population (colonie) est construite à l'aide d'un ACS et l'autre 50% sera produite par un système immunitaire artificiel comme dans l'algorithme original. Le but est de spécialiser les fourmis (solutions) afin de permettre la convergence des fourmis vers l'ensemble PO ou afin de diversifier la recherche de solutions. La Figure 13 présente le pseudo-code d'ACISMOO.

1 : Initialiser l'archive, les phéromones, les visibilités et la colonie
2 : $t \leftarrow 0$
3 : Tant que t < le nombre de cycles maximaux
4 : Pour $k = 0$ à $K/2$ où K est le nombre maximal de fourmis dans la colonie
5 : Initialiser la $fourmi_k$
6 : Pour $i = 0$ à taille du problème
7 : $q \leftarrow$ tirer un nombre aléatoire $\in$ [0,1]
8 : Si $q \leq fourmi_k.q_0$
9 : Choisir j avec une méthode déterministe à l'aide de l'Équation 3.5
10 : Sinon
11 : Choisir j avec une méthode probabiliste à l'aide de l'Équation 3.6
12 : Évaluer $fourmi_k$
13 : Insérer la $fourmi_k$ dans la colonie
14 : Calculer l'assignation de performance de chaque $fourmi_k$ dans la colonie
15 : Trier la colonie par front
16 : Pour chaque $fourmi_k$ du premier front
17 : Calculer le nombre de clones de la $fourmi_k$ à l'aide de l'Équation 3.9
18 : $cpt = 0$
19 : Tant que cpt < le nombre de clones de la $fourmi_k$
20 : Générer deux clones cl_1, cl_2 par copie de la $fourmi_k$
21 : Créer cl_1^{mut} en appliquant la mutation 1 au clone cl_1
22 : Créer cl_2^{mut} en appliquant la mutation 2 au clone cl_2
23 : Évaluer cl_1^{mut} et cl_2^{mut}
24 : Insérer dans la colonie le clone non-dominé
25 : Cpt++
26 : Calculer l'assignation de performance de chaque $fourmi_k$ dans la colonie
27 : Mise à jour des phéromones avec la fourmi ayant le meilleur facteur de dominance et la fourmi ayant le meilleur facteur d'isolement
28 : $Nombre\ de\ fourmis\ archivées \leftarrow$ Insérer chaque $fourmi_k$ de la colonie dans l'archive
29 : $varie \leftarrow$ variation selon le $nombre\ de\ fourmis\ archivées$
30 : Faire varier chaque $fourmi_k$ de la colonie selon $varie$
31 : Mettre les fourmis de la colonie à zéro
32 : t++

Figure 13 : Pseudo-code de l'Ant Colony Immune System for Multiple Objectif Optimization (ACISMOO)

Les lignes 3 à 14 de la Figure 13 représentent la partie où l'ACS construit 50% de la colonie et les lignes 15 à 26 sont la génération de l'autre 50% de la colonie par système immunitaire artificiel.

Dans la partie ACS, une tâche j est choisie à la suite de la tâche i pour une fourmi k selon une règle de transition. Cette règle se retrouve aux lignes 7 à 11 dans la Figure 13. L'ACISMOO dirige sa recherche de j selon une phéromone et deux visibilités qui sont les mêmes que celles utilisées par le BCMC. Également, le seuil (q_0) (ligne 8) qui détermine la règle de transition à utilisée (déterministe ou probabiliste) est unique à chaque fourmi. Un tirage aléatoire est comparé au seuil (q_0) pour déterminer l'utilisation d'une méthode déterministe ou probabiliste pour sélectionner une nouvelle tâche j. La méthode déterministe (exploitation) est définie à l'aide de l'Équation 3.5.

$$j = \max_{h \in S} \left\{ \left(\tau_{ih}(t)\right) * \left(\eta_{ih}^{p}(t)\right)^{\theta} * \left(\eta_{ih}^{f}(t)\right)^{\varpi} \right\} \tag{3.5}$$

où i est la tâche précédente, t représente le cycle présent, τ la trace de phéromones, η_{ih}^{p} et η_{ih}^{f} représente les visibilités liées au premier et deuxième objectif ainsi que θ et ϖ les exposants liés respectivement à l'importance de la visibilité du premier et du second objectif qui sont uniques à chaque fourmi.

La méthode probabiliste (exploration) est définie à l'aide de l'Équation 3.6.

$$p_j(t) = \frac{\left(\tau_{ij}(t)\right) * \left(\eta_{ij}^{p}(t)\right)^{\theta} * \left(\eta_{ij}^{f}(t)\right)^{\varpi}}{\sum_{h \in S} \left(\tau_{ih}(t)\right) * \left(\eta_{ih}^{p}(t)\right)^{\theta} * \left(\eta_{ih}^{f}(t)\right)^{\varpi}} \tag{3.6}$$

Dès que 50% de la colonie est construite, une assignation de performance est réalisée (ligne 14). Cette assignation est empruntée à GISMOO. Le calcul du facteur de dominance est une technique de niching basée sur la notion de dominance au sens Pareto. À chaque itération t, la force $(W(k))$ est calculée pour l'ensemble des fourmis k de la colonie. $W(k)$ est le nombre de solutions y dominées par k :

$$W(k) = |\{y | y \in colonie, k \succ y\}|$$

où $|\ \ |$ représente la cardinalité et $\succ$ la relation de dominance de k par rapport à y. À partir de la force, le facteur de dominance d'une fourmi k, noté $O(k)$, est calculé :

$$O(k) = \begin{cases} \frac{W(k)}{1+2*S(k)}, & Si\ \sum_{y\ \epsilon\ colonie, y\succ k} W(y) = 0 \\ \sum_{y\ \epsilon\ colonie\ y\succ k} W(y), & Sinon \end{cases} \quad (3.7)$$

Quand $\sum_{y\ \epsilon\ colonie, y\succ k} W(y) = 0$, le facteur est inclus entre 0 et 0.5. Le calcul du facteur d'isolement s'inspire de la métrique d'Espacement de Schott (1995) qui évalue la distance séparant une fourmi k de son plus proche voisin (Équation 3.8).

$$Dist(k) = \min_y \left[\sqrt{\left(f_1(k) - f_1(y)\right)^2 + .. + \left(f_n(k) - f_n(y)\right)^2} \right] \quad (3.8)$$

où $n \geq 2$ est le nombre d'objectifs. Les facteurs sont utilisés dans la mise à jour des traces de phéromones et dans la phase immune pour déterminer le nombre de clones à produire pour une fourmi k donnée.

Suite à un tri par front (ligne 15) inspiré de NSGA-II, la phase immune débute. Elle est textuellement reprise de la méthode GISMOO. Le principe de clonage permet de modéliser le fait que seuls les meilleurs anticorps vont proliférer par la suite. Ici, les anticorps correspondent aux fourmis non-dominées de la colonie. La première étape est de trier la colonie en front et de sélectionner les fourmis du premier front comme population d'anticorps (A) à cloner. La seconde étape est de déterminer le nombre de clones à produire par anticorps à l'aide de l'Équation 3.9.

$$nombre\ de\ clones(foumi_k) = round\left[(Dist(fourmi_k) * \frac{K}{2}) \Big/ \sum_{x=1}^{|front_1|} Dist(x) \right] \quad (3.9)$$

où $|front_1|$ est le nombre de fourmis non-dominées du premier front, $Dist(fourmi_k)$ est le facteur d'isolement le la k-ième fourmi et $round$ est une fonction qui retourne le nombre arrondi à l'entier le plus près. Le nombre de clones n'est pas constant et est proportionnel au degré d'isolement de chaque fourmi. Plus une fourmi est isolée, plus le nombre de clones est important. Le clonage est une copie conforme d'une fourmi vers les deux clones. Les clones sont mutés en utilisant des opérateurs de mutation différents pour les deux. La première mutation s'inspire de la structure de voisinage « Exchange neighborhood » utilisée par Arroyo et al (2011) qui se résume en une permutation de deux tâches tirées aléatoirement, tel que le démontre la Figure 14.

	1	2	3	4	5	6	7	8	9
Étape 1	4	8	2	7	9	6	3	5	1
Étape 2	4	8	*3*	7	9	6	*2*	5	1

Figure 14 : Mutation 1 sur une solution de 9 tâches

La mutation se réalise en deux étapes. La première tire deux positions aléatoirement dans la solution. Pour la Figure 14, les positions tirées sont la 3 et la 7, représentées en gris. La deuxième étape est la permutation du contenu des positions, soit dans le présent cas la permutation des tâches 2 et 3 comme indiqué en italique sur la Figure. La deuxième mutation est inspirée de la structure de voisinage « insertion neighborhood » utilisée par Arroyo et al (2011). Elle introduit à un point d'insertion une tâche tirée aléatoirement tel que le démontre la Figure 15.

	1	2	3	4	5	6	7	8	9
Étape 1	4	8	2	7	9	6	3	5	1
Étape 2	4	*6*	2	7	9		3	5	1
Étape 3	4	6	*8*	*2*	*7*	*9*	3	5	1

Figure 15 : Mutation 2 sur une solution de 9 tâches

Cette mutation se réalise en trois étapes. Premièrement, deux positions sont tirées aléatoirement. La première est la position d'accueil, soit la position 2 à la Figure 15, qui est représentée en gris. La seconde est la position de transfert, soit la position 6 de la Figure 15. La deuxième étape consiste à transférer le contenu de la position de transfert vers la position d'accueil, soit dans le présent cas transférer la tâche 6 vers la position 2. Pour la troisième étape, les tâches de la solution d'origine situées entre les positions 2 et 5 inclusivement sont décalées d'une position, tel que le démontrent les tâches en italique à l'étape 3 de la Figure 15. Les clones mutés sont comparés en utilisant la dominance au sens Pareto. Le clone non-dominé remporte le tournoi et est inséré dans la colonie.

La mise à jour des phéromones s'effectue en deux étapes en fin d'itération (ligne 27). En premier lieu, il y a évaporation des traces de phéromones en s'assurant que la trace demeure supérieure à zéro. Dans le cas contraire, la trace est réinitialisée à la valeur initiale. L'évaporation se calcule ainsi : $\tau_{ij} = \tau_{ij} * \rho$ où ρ représente le facteur d'évaporation. En second lieu, le dépôt de phéromones s'effectue par les meilleures fourmis. Dans le présent cas, les meilleures fourmis sont représentées par la fourmi ayant le meilleur facteur de dominance $(bestDom)$ ainsi que la fourmi ayant le meilleur facteur d'isolement $(bestIsol)$. La fourmi $bestDom$ est caractérisée par un facteur de dominance près de 0.5 qui représente que la fourmi a convergée vers le front Pareto. Pour sa part, la fourmi $bestIsol$ est caractérisée par un grand facteur d'isolement qui

représente que la fourmi est éloignée des autres. La recherche des meilleures fourmis s'effectue dans la colonie. Le dépôt de phéromones de la fourmi $bestDom$ se calcule avec l'Équation 3.10 :

$$\tau_{ij} = (1-\rho) * bestDom.facteur + \tau_{ij} \quad (3.10)$$

où $bestDom.facteur$ est le facteur de dominance de la fourmi ayant la meilleure dominance. Pour sa part, le dépôt de la fourmi $bestIsol$ est réalisé avec l'Équation 3.11.

$$\tau_{ij} = (1-\rho) * RatioIsol + \tau_{ij} \quad (3.11)$$

où $RatioIsol$ est le ratio de l'isolement entre la première et la seconde fourmi ayant les meilleurs facteurs d'isolement.

La variation de la fourmi, réalisée de la ligne 28 à 30, s'effectue sur les paramètres dynamiques: le seuil (q_0) de la règle de transition ainsi que sur les exposants liés à l'importance de la visibilité du premier objectif (θ) et du second objectif (ϖ). Le nombre de fourmis insérées dans l'archive au cours d'un cycle est comptabilisé. Plus il y a de fourmis entrées dans l'archive et moins les paramètres dynamiques varient. Puisque la configuration des paramètres actifs produit des solutions (fourmis) de bonne qualité, il semble opportun de la conserver inchangée. La variation s'effectue autant sur le nombre de paramètres dynamiques que sur la quantité positive ou négative qu'un paramètre dynamique fluctue. La variation est appliquée après chaque cycle.

Les résultats d'ACISMOO sont comparés au BCMC à la Section 4.2.2. Il est à noter qu'ACISMOO est une méthode uni-colonie inspirée de GISMOO. Elle propose, également, la mise à jour des phéromones basée sur les fourmis ayant le meilleur facteur de dominance et le meilleur facteur d'isolement, ce qui n'a pas été observé dans la

littérature. Pour permettre de diriger la recherche, ACISMOO propose la variation du seuil (q_0) et des exposants liés à l'importance de visibilité.

3.5 Amélioration d'ACISMOOmodif

La version d'ACISMOO présentée à la Section 3.4.2 a été réalisée dans le but d'être une méthode proposant une comparaison équitable avec BCMC. La présente section propose des améliorations à ACISMOO au niveau de la visibilité, de l'emplacement dans l'algorithme pour calculer l'évaluation de la fourmi (OTA) ainsi qu'une amélioration au niveau de la variation des fourmis. La variation des fourmis est vue ici comme étant la modification à la fin de chaque cycle des paramètres dynamiques au sein des fourmis. L'intention derrière ces améliorations est de rendre les fourmis de la colonie plus spécialisées, ce qui va permettre d'augmenter leur convergence ou leur diversification.

L'*amélioration de la visibilité* se situe au niveau de la visibilité du premier objectif : la visibilité liée aux pénalités d'avance et de retard. L'initialisation du déterminant demeure telle qu'expliqué à la Section 3.4.1. La différence est qu'un déterminant considère plusieurs visibilités. Une telle considération permet à deux fourmis avec le même déterminant de sélectionner des tâches différentes, ce qui apporte de la diversité. Les prochains paragraphes définissent les déterminants ainsi que les visibilités considérées.

Le déterminant $avance$ est initialisé quand l'ensemble des tâches à ordonnancer à une position j dans la fourmi k sont en avance. Dit autrement, l'ensemble des tâches disponibles à la position j dans la fourmi k seront ordonnancées avant leur fenêtre d'échéance. Il est de rigueur de rappeler que le calcul de l'avance (E) est effectué tel que décrit l'Équation 3.12

$$E = \max \{0;\ (de_j - C_j\} \quad (3.12)$$

et le calcul de la pénalité d'avance est présenté à l'Équation 3.13.

$$pénalité\ d'avance = E * \alpha \quad (3.13)$$

Étant donné que peu importe la tâche ordonnancée à la position j, elle sera en avance, alors le but est de minimiser l'impact de la pénalité de l'avance. Pour ce faire, selon les Équations 3.12 et 3.13, il est essentiel de diminuer certaines composantes des équations : l'avance (E), le coût de la pénalité d'avance (α) ou les deux. Un ensemble de visibilités orientées avance est proposé, qui permettent de diminuer une ou plusieurs composantes énumérées. Le Tableau 6 présente l'ensemble de ces visibilités orientées avance.

Visibilité	**Explication**
$\frac{p_j + s_{ij}}{\alpha_j}$	Privilégie la tâche j ayant un petit coût de pénalité et un grand temps d'exécution
$\frac{1}{\alpha_j}$	Privilégie la tâche j ayant le plus petit coût de pénalité
$p_j + s_{ij}$	Privilégie la tâche j ayant le plus grand temps d'exécution
$1 / \max\{0; (de_j - C_j)\}$	Privilégie la tâche j ayant la plus petite marge d'avance

Tableau 6 : Ensemble des visibilités orientées avance

La première colonne donne la visibilité et la seconde explique quelle tâche la visibilité va privilégier. Tel que le démontre le Tableau 6, certaines visibilités tendent à diminuer le coût de la pénalité d'avance (α). Cette diminution a un impact direct sur le calcul de la pénalité d'avance de l'Équation 3.13. D'autres visibilités vont plutôt privilégier les grands temps d'exécution, ceci permet de diminuer l'impact de la pénalité d'avance car elles amoindrissent le calcul de l'avance (E) de l'Équation 3.12.

Le déterminant $retard$ est initialisé quand la majorité des tâches à une position j dans une fourmi k sont en retard. Le déterminant $retard$ propose de sélectionner à l'aide des visibilités orientées retard des tâches qui diminuent l'impact de l'insertion d'une tâche en retard à la position j dans la fourmi k sur les autres tâches qui restent à ordonnancer ou bien de minimiser la pénalité de retard. Pour ce faire, le Tableau 7 présente l'ensemble des visibilités orientées retard.

Visibilités	**Explication**
$\beta_j / p_j + s_{ij}$	Privilégie la tâche j ayant une grande pénalité de retard et un petit temps d'exécution
β_j	Privilégie la tâche j ayant une grande pénalité de retard
$1 / p_j + s_{ij}$	Privilégie la tâche j ayant un petit temps d'exécution
$1 / \max\{0; (C_j - dt_j)\}$	Privilégie la tâche j ayant une petite marge de retard
$\max\{0; (C_j - dt_j)\}$	Privilégie la tâche j ayant une grande marge de retard

Tableau 7 : Ensemble des visibilités orientées retard

Le Tableau 7 est sous la même forme que le Tableau 6. Tel que le démontre le Tableau 7, certaines visibilités privilégient un grand coût de pénalité (β). La pénalité de retard n'est pas diminuée immédiatement, mais cela évite de repousser une tâche en retard qui a un coût de pénalité de retard (β) élevé. D'autres visibilités vont privilégier les petits temps d'exécution. Cette façon de faire a deux justifications : elle évite d'augmenter le retard des autres tâches et permet de diminuer le calcul de la pénalité en réduisant le retard (T).

Le déterminant à $temps$ est initialisé quand la majorité des tâches à la position j pour la fourmi k sont à temps. Autrement dit, la majorité des tâches sont dans leur fenêtre

d'échéance ($de_j \leq C_j \leq dt_j$). Il est à noter qu'une tâche insérée dans sa fenêtre d'échéance n'entraine aucune pénalité d'avance ou de retard. L'intention du déterminant à $temps$ est alors de favoriser la sélection de tâches qui vont permettre aux autres d'être insérées dans leur fenêtre d'échéance. Cela permet également d'ordonnancer le plus rapidement possible des tâches qui pourraient devenir en retard. Le Tableau 8 présente des visibilités orientées à temps.

Visibilités	**Explication**
$1/p_j + s_{ij}$	Privilégie la tâche j ayant le temps d'exécution le plus court
$\frac{\beta_j}{\alpha_j}$	Privilégie la tâche j ayant le plus grand écart entre β et α
$1/\max\{0; (dt_j - C_j)\}$	Privilégie la tâche j ayant la plus petite marge avant d'être en retard

Tableau 8 : Ensemble des équations pour le déterminant à $\boldsymbol{temps}$

Tel que le démontre le Tableau 8, certaines visibilités privilégient les petits temps d'exécution, ce qui permet aux autres tâches d'avoir la chance d'être insérées dans leur fenêtre d'échéance. D'autres visibilités vont privilégier un coût de pénalité de retard élevé. Cette stratégie permet d'ordonner une tâche qui a un coût de pénalité de retard élevé, donc il est sûr qu'elle ne sera pas en retard.

L'amélioration d'ACISMOO qui est maintenant présentée est *l'emplacement de l'évaluation de la fourmi (OTA)*. L'ACISMOO présenté à la Section 3.4.2 et BCMC évaluent la fourmi quand celle-ci est entièrement construite. La faiblesse de cet emplacement est qu'OTA est un algorithme qui peut modifier considérablement les dates de fin des tâches ordonnancées en raison de l'insertion de temps mort. L'ordonnancement

des tâches d'une fourmi qui semblent adéquate peut devenir désastreux après OTA. Toutefois, les résultats de ACISMOO de la Section 3.4.2 portent à croire qu'évaluer la fourmi après la construction génère quand même de bons résultats. Malgré ce point, la visibilité liée à la pénalité d'avance et de retard est basée en partie sur le temps d'exécution total (C_j) de la fourmi à un moment précis. De ce fait, en évaluant la fourmi avec OTA après sa construction, la sélection d'une tâche j à l'aide de la règle de transition utilise une approximation de C_j. Avec ces points en main, une amélioration est proposée dans ce mémoire. L'évaluation de la fourmi peut être réalisée à deux emplacements différents : après la construction de la fourmi (comme ACISMOO de la Section 3.4.2 et BCMC) ou après l'insertion d'une tâche dans la fourmi. Ce dernier emplacement permet d'avoir un C_j exact à chaque fois qu'une tâche est sélectionnée. Tel que dit précédemment, l'évaluation à la fin de la construction génère somme toute de bons résultats, donc chaque fourmi de la colonie a un emplacement pour l'évaluation qui est donné aléatoirement en début d'algorithme. L'emplacement donné à une fourmi ne varie pas au travers des cycles.

La dernière amélioration apportée à ACISMOO est la *variation des fourmis*. La phase de variation des fourmis demeure à la même place dans l'algorithme, soit à la fin de chaque cycle. Également, la variation d'une fourmi est dirigé par le nombre de solutions non-dominées insérées dans l'archive. L'amélioration proposée, dans cette section, est d'ajouter de nouveaux paramètres dynamiques à la fourmi. Ces nouveaux paramètres dynamiques représentent quelle visibilité est utilisée pour chaque déterminant. Des fourmis avec un même déterminant peuvent ainsi avoir une visibilité différente. Lors de la phase de variation, la visibilité peut être modifiée. Donc les paramètres dynamiques qui sont sujets à être modifiés pour chaque fourmi sont dorénavant : le seuil de la règle de

transition (p_0), les exposants liées à l'importance de visibilité θ et ϖ ainsi que les visibilités pour chaque déterminant.

CHAPITRE 4

EXPÉRIENCES NUMÉRIQUES ET RÉSULTATS

4.1 Conditions d'expérimentation

Les méthodes de résolution ont été implémentées en C++ avec Visual Studio 2008 sur un ordinateur ayant un processeur Intel Core i-7-2600 @3.40ghz avec 12 go de RAM. Chacune des méthodes est exécutée cinq fois par instance pour plus de robustesse. Dans la même optique, le résultat maximal des cinq exécutions est conservé pour l'analyse. Les paramètres d'exécution sont divisés en trois catégories : ceux pour les méthodes NSGA-II, PMS^{MO} et GISMOO, ceux pour la méthode BCMC ainsi que ceux pour les méthodes ACISMOO et ACISMOOmodif. Toutefois, le critère d'arrêt représenté par le nombre d'évaluation est déterminé à 100 000 et les mutations sont les mêmes pour l'ensemble des méthodes.

Les paramètres utilisées pour NSGA-II, PMS^{MO} et GISMOO sont déterminés en tenant compte des recommandations des auteurs respectifs : la probabilité de mutation à 0,06, la taille de la population est de 100 individus et la méthode de croisement est le Random Maximal Preservative Crossover (RMPX) (Sioud et al. 2009). Le RMPX est démontré performant sur le problème d'expérimentation et ce dernier se rapproche du problème MURMO.

Les paramètres de BCMC sont fixés suivant les recommandations de l'article (Iredi et al. 2001) : 0,2 pour le facteur de persistance (ψ), 1 pour les exposants liés à l'importance de la trace de phéromones et de la visibilité $(\delta \; et \; \varepsilon)$, la trace de phéromones initiale (τ_0) est de 1,0, le nombre de colonies est de 10 et le nombre total de fourmis est égal au nombre de tâches. Iredi et al (2001) initialisent leur nombre de fourmis à 100, les instances de

problème traitées ne sont que de 100 tâches, donc il en a été déduit que le nombre de fourmis total est égal au nombre de tâches du problème.

Pour terminer, les paramètres utilisés pour ACISMOO et ACISMOOmodif sont déterminés en se basant sur les recommandations de l'ACS uni-objectif (Dorigo et al. 1997) : 0,9 pour le facteur d'évaporation (ρ), le seuil de la règle de transition (q_0) est compris entre 0,1 et 0,9, les exposants liés à l'importance de la visibilité du premier objectif (θ) et du deuxième objectif (ϖ) sont compris entre 1 et 5, la trace de phéromones initiale (τ_o) est de 0,01 et le nombre total de fourmis est de 20. La phase immune d'ACISMOO demande deux mutations. Les mutations utilisées sont inspirées des structures de voisinage du MOVNS3 (Arroyo et al. 2011). La première mutation, présentée à la Section 3.4.2, est utilisée dans NSGA-II, PMS^{MO}, GISMOO, ACISMOO et ACISMOOmodif. La deuxième mutation est utilisée dans GISMOO, ACISMOO et ACISMOOmodif. Pour finir, les 72 instances utilisées sont celles offertes par Arroyo et al (2011).

4.1.1. Les instances

La taille des instances est le nombre de tâches à ordonnancer $m = 20, 30, 40, 50, 75\ et\ 100$. Chaque tâche j a un temps d'exécution p_j et une pénalité de retard β_j générés uniformément dans l'intervalle $[1, 100]$ et $[20, 100]$, respectivement. Généralement, les tâches ordonnancées en retard sont plus néfastes que celles en avance, alors la pénalité d'avance α_j est générée en multipliant k à la pénalité de retard (β_j) où k est un nombre tiré aléatoirement entre $[0, 1]$. Le centre de la fenêtre d'échéance $[de_j, dt_j]$ est généré uniformément entre $[(1 - R - RDD/2)TP,\ (1 - R + RDD/2)TP]$ où TP est la somme des temps d'exécution des tâches, R est le facteur de retard et RDD est la taille

relative de la fenêtre d'échéance. R et RDD prennent respectivement les valeurs {0,1, 0,2, 0,3, 0,4} et {0,8, 1,0, 1,2}. La grandeur de la fenêtre d'échéance est uniformément générée entre $[1, TP/m]$. Pour toutes les paires de tâches (i, j), si $i \neq j$ alors un temps de réglage s_{ij} est généré uniformément dans l'intervalle $[0, 50]$. R et RDD prennent respectivement 4 et 3 valeurs différentes, donc le total des instances avec ces deux paramètres est de 12. Étant donné qu'une instance est générée avec une paire de (R, RDD) et qu'il y a 6 catégories de tâches $(20, 30, 40, 50, 75, 100)$, le nombre d'instances créées au total est de 12 X 6 = 72 (Arroyo et al. 2011).

4.2 Résultats et analyse

4.2.1. Comparaison globale des solutions des méthodes adaptées

Plusieurs méthodes de résolution ont été adaptées pour le problème MURMO dans ce mémoire. Le Tableau 9 présente l'Hypervolume sous forme de trois groupes. Le premier est la méthode qui provient de la littérature dont les résultats ont été offerts par les auteurs, le MOVNS3 (Arroyo et al. 2011). Le deuxième groupe est l'ensemble de méthodes adaptées au problème MURMO dans l'optique de confectionner une collection de solutions pour comparaison future. Le troisième groupe est les approches d'OCF adaptées pour ce mémoire.

	MOVNS3 (Arroyo et al. 2011)	NSGAII	PMSMO	GISMOO	ACISMOO	ACISMOO modif	BCMC
20	1,82E+09	2,78E+09	2,90E+09	2,96E+09	2,05E+09	2,53E+09	1,55E+09
30	1,88E+11	2,03E+11	2,03E+11	2,06E+11	4,72E+10	2,04E+11	4,28E+10
40	2,12E+10	3,72E+10	3,71E+10	4,27E+10	2,33E+10	3,13E+10	1,31E+10
50	4,56E+10	8,85E+10	8,54E+10	1,03E+11	5,46E+10	7,24E+10	2,75E+10
75	1,88E+11	4,25E+11	4,06E+11	5,20E+11	2,37E+11	3,63E+11	1,12E+11
100	4,48E+11	1,29E+12	1,24E+12	1,66E+12	7,67E+11	1,15E+12	3,55E+11

Tableau 9 : Métrique Hypervolume de l'ensemble des méthodes adaptées dans ce mémoire

Les lignes du Tableau 9 représentent les groupes d'instances en fonction de la taille. En colonnes, il y a les méthodes. La moyenne des Hypervolumes se retrouve à la jonction d'une ligne et d'une colonne. Un bon Hypervolume est caractérisé par une grande aire sous la courbe de l'ensemble PO. L'analyse du Tableau 9 démontre que GISMOO (en gris) a un meilleur Hypervolume parmi toutes les méthodes. BCMC et MOVNS3 sont près l'une de l'autre en termes d'Hypervolume. ACISMOOmodif est légèrement supérieur à ACISMOO. Aussi, ACISMOOmodif a un Hypervolume similaire à celui de GISMOO.

Pour permettre une bonne lisibilité de la métrique Couverture (Tableau 10), trois méthodes sont retenues du Tableau 9. Le choix des méthodes est guidé par les performances des méthodes à la métrique Hypervolume et l'origine des méthodes. Les méthodes sont : MOVNS3 car elle est la méthode proposée dans le travail original de présentation du problème MURMO (Arroyo et al. 2011), GISMOO car elle est la meilleure méthode connue pour résoudre le problème MURMO et ACISMOOmodif car c'est la

méthode parmi les approches d'OCF ayant eu la meilleure performance pour l'Hypervolume.

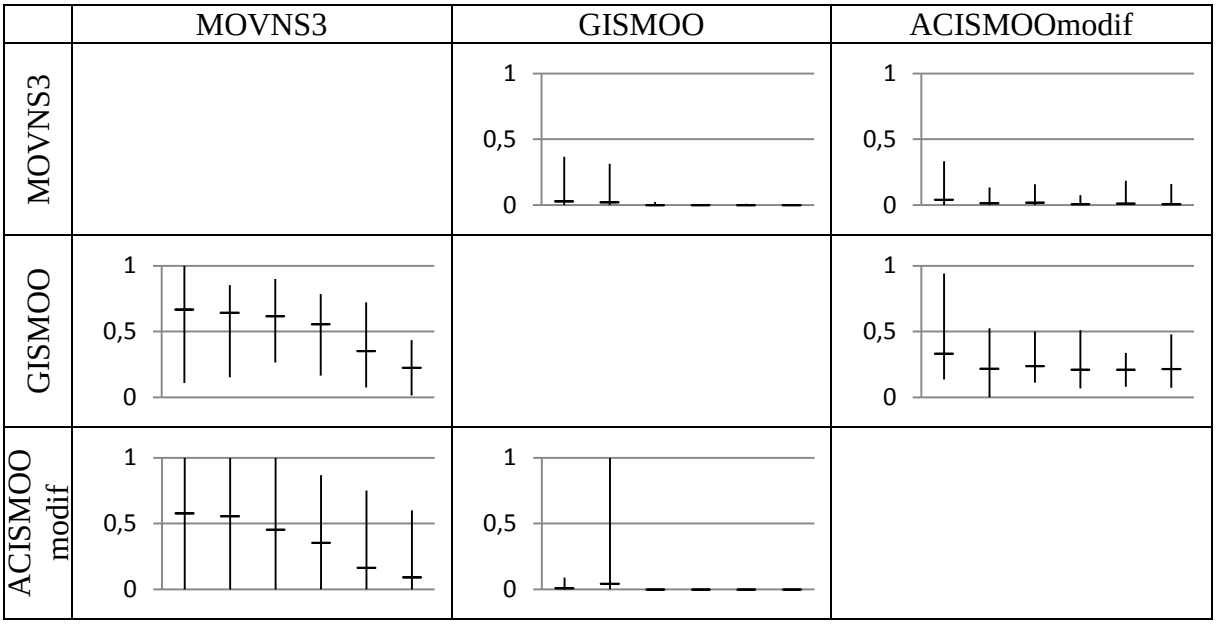


Tableau 10 : Métrique Couverture de MOVNS3, GISMOO et ACISMOOmodif

La métrique Couverture compare deux à deux les méthodes. Le Tableau 10 compare les méthodes en lignes avec celles en colonnes. Chacune des lignes verticales dans les graphiques du Tableau 10 représente les groupes d'instances en fonction de la taille $20, 30, 40, 50, 75\ et\ 100$ tâches. Les lignes horizontales représentent le pourcentage de solutions de la méthode en ligne qui dominent au moins une solution de la méthode en colonne. L'analyse du Tableau 10 permet de constater que GISMOO et ACISMOOmodif ont plus de solutions qui dominent MOVNS3 que l'inverse. Cette constatation confirme l'Hypervolume supérieure de GISMOO et d'ACISMOOmodif par rapport à MOVNS3.

La métrique traitant de la distribution des solutions est présentée par la suite pour compléter la comparaison entre MOVNS3, GISMOO et ACISMOOmodif. La distribution des solutions d'un ensemble PO permet de valider si les solutions sont réparties tout le long du front Pareto ou bien concentrées dans un secteur. La métrique choisie est l'Espacement, qui mesure l'uniformité de la répartition des solutions de l'ensemble PO (Collette et al. 2002). Un bon résultat est celui qui est minimal. Le Tableau 11 présente la métrique Espacement entre MOVNS3, GISMOO et ACISMOOmodif.

	MOVNS3	GISMOO	ACISMOOmodif
20	6 040	1 937	5 578
30	7 170	5 141	15 270
40	7 170	4 541	36 820
50	69 932	6 915	52 799
75	28 949	23 296	74 010
100	377 747	38 820	251 054

Tableau 11 : Métrique Espacement pour les méthodes MOVNS3, GISMOO et ACISMOOmodif

Le Tableau 11 est sous la même forme que le Tableau 9, à la différence que la jonction entre une ligne et une colonne du Tableau 11 représente la moyenne de la métrique Espacement des groupes d'instances et la méthode en gris représente celle ayant le mieux réussi. Le résultat de la métrique Espacement est généralement près de zéro. Les résultats élevés démontrent que les ensembles PO sont disjoints. Toutefois, il est possible de remarquer que GISMOO a une meilleure répartition des solutions. Également, ACISMOO et MOVNS3 s'échangent la deuxième meilleure répartition selon les groupes d'instances.

La nature bi-objectifs du problème MURMO permet de représenter sur un plan les ensembles PO. Cette représentation fournit un appui visuel aux décideurs. Elle aide à comprendre les métriques discutées précédemment. La Figure 16 est une représentation de l'ensemble PO d'une instance typique.

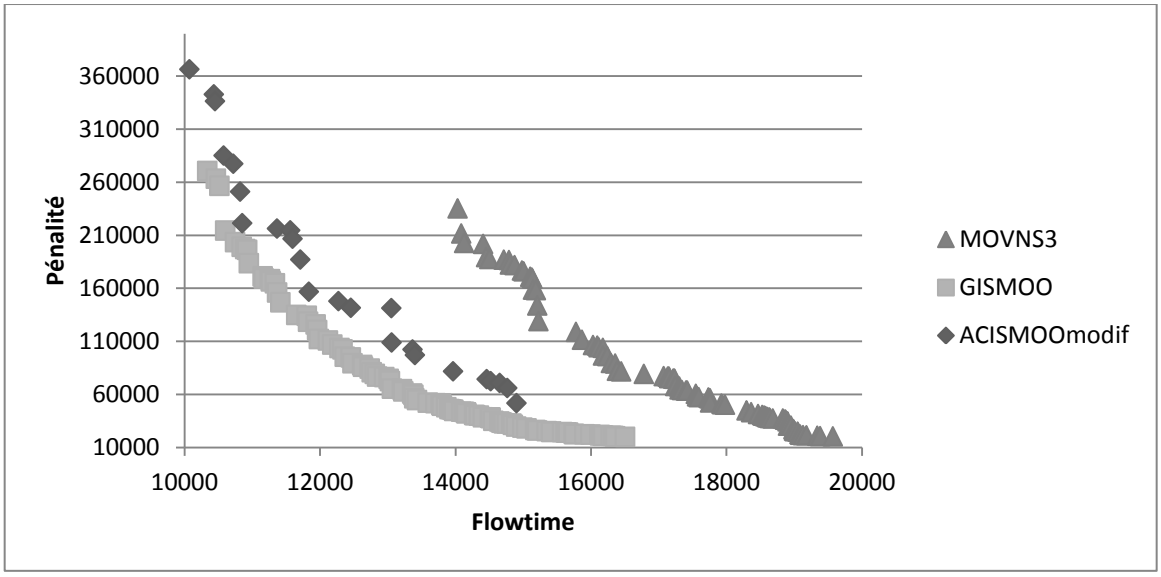


Figure 16 : Illustration des ensembles PO d'une instance typique pour les méthodes MOVNS3, GISMOO et ACISMOOmodif (30 tâches)

L'abscisse représente l'objectif lié au flowtime et l'ordonnée représente l'objectif lié à la pénalité d'avance et de retard. Cette représentation illustre que GISMOO génère des résultats de meilleure qualité que MOVNS3 et ACISMOOmodif car sa courbe est plus près de l'origine. Le graphique permet de visualiser l'impact du calcul de la fonction objectif. OTA privilégie l'objectif lié à la pénalité d'avance et de retard au détriment de l'objectif lié au flowtime car il tente d'insérer les tâches dans leur fenêtre d'échéance en incluant des temps morts. MOVNS3 génère des solutions qui favorisent fortement l'objectif lié à la pénalité d'avance et de retard. Toutefois, GISMOO et ACISMOOmodif introduisent des

mécanismes qui permettent de trouver des solutions qui favorisent l'objectif lié au flowtime. Particulièrement, ACISMOOmodif sélectionne dans sa règle de transition les tâches ayant, généralement, un court temps d'exécution. Une telle tendance privilégie le flowtime (Pinedo 2012). Au niveau de la distribution des solutions, la Figure 16 permet de visualiser la situation. Il est possible de constater que les trois ensembles PO sont disjoints. L'ensemble PO généré par GISMOO a des solutions très rapprochées les une des autres, d'où sa performance dans la métrique Espacement. Malgré les efforts de diversité, ACISMOOmodif ne réussit pas à couvrir autant que GISMOO.

4.2.2. Comparaison BCMC et ACISMOO

Cette comparaison porte sur une approche OCF multi-colonies, le BCMC, avec une approche OCF uni-colonie, l'ACISMOO. L'analyse des ensembles PO à l'aide de métriques va permettre de voir les forces et faiblesses des deux approches. Cette section présente en premier les métriques évaluant la qualité des solutions des ensembles PO. En deuxième, à la Section 4.2.2.2 présente la métrique évaluant la distribution des solutions des ensembles PO. Étant donné la nature bi-objectifs du problème MURMO, il possible de représenter sur un plan les ensembles PO.

4.2.2.1. Métriques mesurant la qualité des solutions des ensembles PO

La qualité des solutions d'un ensemble PO par rapport à un autre permet de connaitre à quel point une méthode a généré des solutions qui ont convergé vers le front Pareto. La première métrique analysée est l'Hypervolume. Le Tableau 12 présente la comparaison entre l'Hypervolume d'ACISMOO et de BCMC.

	ACISMOO	BCMC
20	2,05E+09	1,55E+09
30	4,72E+10	4,28E+10
40	2,33E+10	1,31E+10
50	5,46E+10	2,75E+10
75	2,37E+11	1,12E+11
100	7,67E+11	3,55E+11

Tableau 12 : Métrique Hypervolume pour les méthodes ACISMOO et BCMC

Les meilleurs résultats sont indiqués en gris. ACISMOO présente une meilleure aire sous la courbe que le BCMC, pour le problème MURMO. L'écart entre ACISMOO et BCMC augmente avec le nombre de tâches des instances.

La seconde métrique analysée est la Couverture, qui permet également de déterminer la qualité des solutions d'une méthode par rapport à une autre. Le Tableau 13 compare les méthodes en ligne avec celles en colonne.

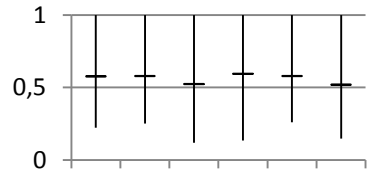
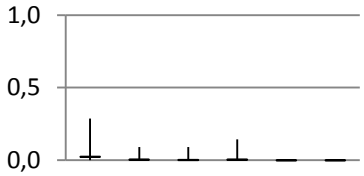
	ACISMOO	BCMC
ACISMOO		
BCMC		

Tableau 13 : Métrique Couverture pour les méthodes ACISMOO et BCMC

Le Tableau 13 est sous la même forme que le Tableau 10, à la différence qu'il est pour deux méthodes. En moyenne, ACISMOO domine 56% des solutions de BCMC. Une telle dominance d'ACISMOO confirme les résultats de l'Hypervolume. Les résultats de ces deux métriques permettent de conclure que l'utilisation de concepts AE Pareto génère des solutions de meilleure qualité.

4.2.2.2. Métriques mesurant la distribution des solutions des ensembles PO

Un ensemble PO est évalué selon qu'il converge vers le front Pareto (qualité) et selon la répartition des solutions sur la frontière. Le Tableau 14 présente la métrique Espacement qui estime la distribution des solutions pour un ensemble PO.

	ACISMOO	BCMC
20	10 933	32 110
30	22 579	16 858
40	36 499	45 903
50	32 773	85 841
75	67 989	197 358
100	132 156	308 771

Tableau 14 : Métrique Espacement pour les méthodes ACISMOO et BCMC

Le Tableau 14 est sous la même forme que le Tableau 11. La première constatation est que les deux ensembles PO sont disjoints car la mesure de l'Espacement est loin de zéro. Généralement, la répartition des solutions d'ACISMOO est meilleure que BCMC. Rappelons qu'ACISMOO a un nombre total de fourmis fixe à 20 et BCMC a un nombre total de fourmis qui varie selon le groupe d'instance et est basé sur le nombre de tâches. Avec le même nombre total de fourmis, soit le groupe à 20 tâches, ACISMOO a une

meilleure répartition. Ceci démontre que les mécanismes de diversité utilisés dans ACISMOO, tel que les paramètres dynamiques des fourmis et le nombre de clones selon le facteur d'isolement permettent de distribuer les solutions sur le front Pareto. Pour sa part, BCMC base essentiellement sa diversité sur le nombre total de fourmis, ce qui apporte peu de performance pour le problème MURMO.

4.2.2.3. Représentation graphique

Lorsque le problème le permet, représenter les ensembles PO sur un plan permet d'avoir une conceptualisation des métriques, tel que le démontre la Figure 17.

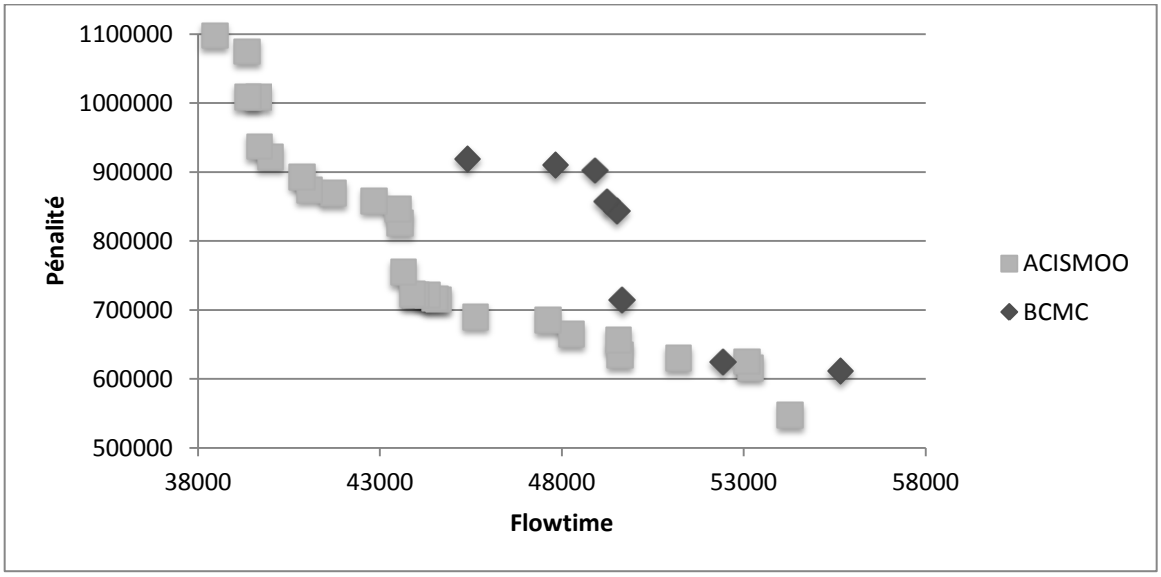


Figure 17 : Illustration des ensembles PO d'une instance typique pour les méthodes ACISMOO et BCMC (40 tâches)

La Figure 17 est sous la même forme que la Figure 16 à la différence d'être pour deux méthodes. La Figure 17 donne une image fidèle des métriques étudiées précédemment. Il est possible de remarquer qu'ACISMOO génère des solutions de meilleure qualité que BCMC. Également, les deux ensembles PO sont disjoints. ACISMOO couvre mieux le front Pareto que BCMC. Aussi, ACISMOO trouve plus de solutions non-dominées que BCMC.

4.2.3. Amélioration ACISMOO

Tel que présenté à la Section 3.5, les améliorations de ACISMOO ont pour but de rendre la méthode plus performante en terme de qualité et de distribution des solutions. La comparaison s'effectue entre ACISMOO et les améliorations, appelées ACISMOOmodif. Cette section présente d'abord les métriques évaluant la qualité des solutions des ensembles PO, puis celles mesurant la distribution des solutions des ensembles PO. Par la suite, une sous-section présente l'analyse d'une illustration planaire d'ensembles PO.

4.2.3.1. Métriques mesurant la qualité des solutions des ensembles PO

Deux métriques sont étudiées dans cette section : l'Hypervolume et la Couverture. Le Tableau 15 présente l'Hypervolume d'ACISMOO et d'ACISMOOmodif.

	ACISMOO	ACISMOOmodif
20	2,05E+09	2,53E+09
30	4,72E+10	2,04E+11
40	2,33E+10	3,13E+10
50	5,46E+10	7,24E+10
75	2,37E+11	3,63E+11
100	7,67E+11	1,15E+12

Tableau 15 : Métrique Hypervolume pour les méthodes ACISMOO et ACIMOOmodif

L'Hypervolume le plus élevé est le plus performant et est représenté en gris dans le Tableau 15. ACISMOOmodif possède une meilleure aire sous la courbe que ACISMOO, dit autrement ACISMOOmodif améliore la qualité des solutions selon la métrique Hypervolume. Tel que constaté lors de la comparaison entre ACISMOO et BCMC, l'écart entre l'Hypervolume d'ACISMOOmodif et ACISMOO s'accroit avec l'augmentation du nombre de tâches. Cette amélioration est aussi visible sur la métrique Couverture, tel que démontré par le Tableau 16.

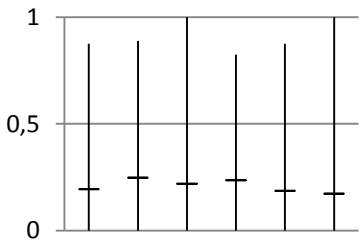
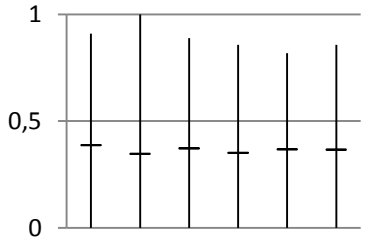
	ACISMOO	ACISMOOmodif
ACISMOO		
ACISMOOmodif		

Tableau 16 : Métrique Couverture pour les méthodes ACISMOO et ACISMOOmodif

Les solutions d'ACISMOOmodif dominent en moyenne 37% des solutions d'ACISMOO. À l'inverse, les solutions d'ACISMOO dominent en moyenne 21% des solutions d'ACISMOOmodif. La Couverture confirme donc les résultats de l'Hypervolume. De tels résultats sont possibles essentiellement grâce à la modification de l'emplacement de l'évaluation de la fourmi. Quand celle-ci est évaluée de façon incrémentale, le temps d'exécution total (C_j) de la dernière tâche insérée tient compte de la possibilité d'avoir des temps morts. La qualité des solutions générées est supérieure car la sélection d'une tâche est réalisée à l'aide de la règle de transition qui est basée sur le temps d'exécution total (C_j). Cette sélection est alors plus précise.

4.2.3.2. Métriques mesurant la distribution des solutions des ensembles PO

La section précédente démontre qu'ACISMOOmodif améliore la qualité des solutions. La prochaine métrique évalue la répartition des solutions pour déterminer à quel point les améliorations permettent de distribuer les solutions sur le front Pareto. Le Tableau 17 présente les résultats de la métrique Espacement.

	ACISMOO	ACISMOOmodif
20	10 933	5 578
30	22 579	15 270
40	36 499	36 820
50	32 773	52 799
75	67 989	74 010
100	132 156	251 054

Tableau 17 : Métrique Espacement pour les méthodes ACISMOO et ACISMOOmodif

Le Tableau 17 démontre qu'ACISMOO répartie mieux ses solutions le long du front Pareto qu'ACISMOOmodif pour les groupes d'instances de 40, 50, 75 et 100 tâches. Toutefois, les groupes d'instances de 20 et 30 tâches d'ACISMOOmodif améliorent la répartition des solutions. Rappelons que la répartition est la distance entre les solutions d'un ensemble PO. Par conséquence, pour les groupes d'instance 40, 50, 75 et 100 tâches, ACISMOOmodif génère des ensembles PO avec les distances entres les solutions plus grandes qu'ACISMOO. Cette distance peut être due à un manque de diversité ou un manque d'intensification. La représentation graphique des ensembles PO des méthodes va permettre de statuer sur les raisons de la distance entre les solutions de l'ensemble PO d'ACISMOOmodif.

4.2.3.3. Représentation graphique

La Figure 18 illustre les ensembles PO d'une instance typique qui va permettre d'expliquer à partir d'un plan les constats des métriques.

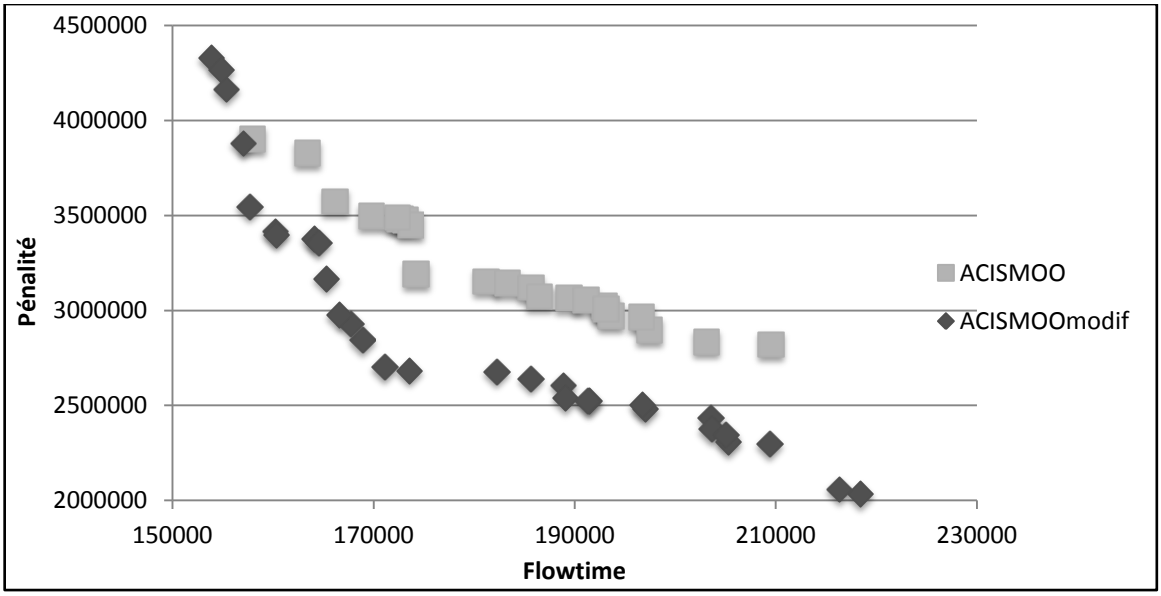


Figure 18 : Illustration des ensembles PO d'une instance typique pour les méthodes ACISMOO et ACISMOOmodif (75 tâches)

Le plan de la Figure 18 démontre qu'ACISMOOmodif génère des solutions de meilleure qualité qu'ACISMOO car le front Pareto d'ACISMOOmodif est situé en dessous de celui d'ACISMOO. Également, la Figure 18 permet de voir la répartition des solutions. Les solutions d'ACISMOO sont plus près les unes des autres que les solutions d'ACISMOOmodif, ce qui confirme la métrique Espacement. Étant donné que les solutions d'ACISMOOmodif sont distancées entre elles, il est possible de conclure que la méthode diversifie sa recherche mais ne l'intensifie pas assez. Cette constatation est remarquable par

le front Pareto d'ACISMOOmodif qui contient plusieurs interstices. Toutefois, le front Pareto d'ACISMOOmodif est beaucoup plus large qu'ACISMOO. Cette différence démontre qu'ACISMOOmodif diversifie mieux sa recherche qu'ACISMOO. De plus, ACISMOOmodif trouve des solutions de compromis qu'ACISMOO ne trouve pas ce qui en fait un portrait plus intéressant pour un décideur.

4.3 Observations liées à la conception des instances

Tel que vu à la Section 4.1.1, les instances sont définies entre autres par le facteur de retard R et la taille relative de la fenêtre RDD. Les instances sont numérotées de 1 à 12 où chaque numéro représente un facteur R et une taille RDD différents. Le Tableau 18 présente la distribution du facteur de retard (R) et de la taille relative de la fenêtre d'échéance (RDD) selon la numérotation des instances.

Numéro d'instance	R	RDD
1	0.1	0.8
2	0.1	1.0
3	0.1	1.2
4	0.2	0.8
5	0.2	1.0
6	0.2	1.2
7	0.3	0.8
8	0.3	1.0
9	0.3	1.2
10	0.4	0.8
11	0.4	1.0
12	0.4	1.2

Tableau 18 : Distribution du R et du RDD dans les instances

Tel que le montre le Tableau 18, le facteur R est minimal aux instances 1, 2 et 3 et il est maximal aux instances 10, 11, et 12. Pour permettre d'illustrer la différence entre un facteur R minimal et maximal également étant donné que le problème le permet, les instances sont représentées sur un graphique planaire. Pour ce faire, les méthodes choisies sont les mêmes que la Section 4.2.1 : MOVNS3 (▲), GISMOO (●) et ACISMOOmodif(■).

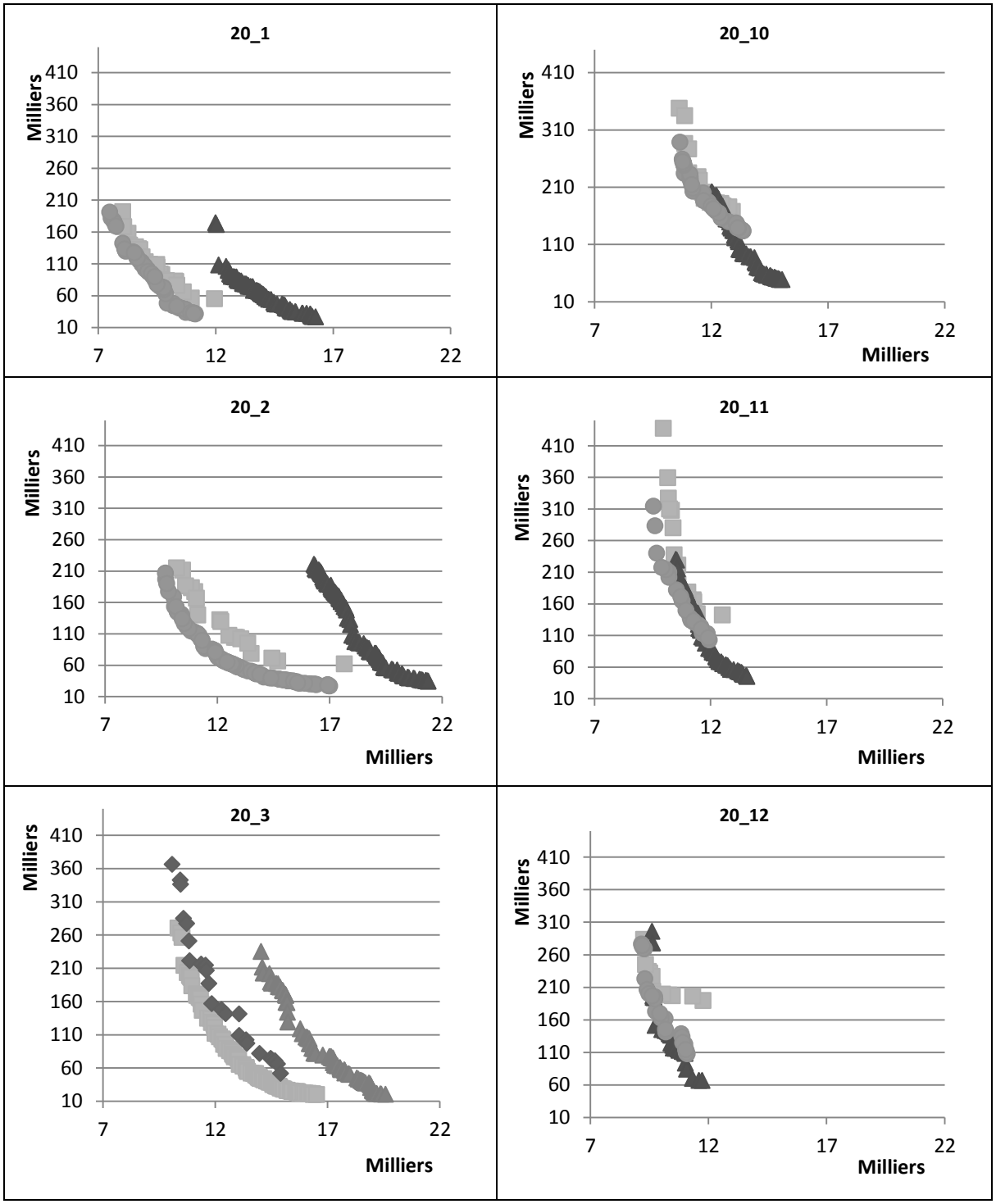


Figure 19 : Illustration des fronts Pareto générés par MOVNS3, GISMOO et ACISMOOmodif

Chaque graphique de la Figure 19 est sous la même forme que la Figure 16. Le titre de chaque graphique donne le nombre de tâches et le numéro de l'instance. La variation du factuer R a pour effet que les fronts Pareto générés sont plus regroupés sur les instances 10, 11 et 12 (deuxième colonne de la Figure 19) que sur les instances 1, 2 et 3 (première colonne de la Figure 19). La raison de ce constat est que le facteur de retard (R) des instances est une des variables qui déterminent les bornes du centre de la fenêtre d'échéance. En analysant les extrêmes de cette borne, un facteur R de 0,1 donne des bornes qui sont éloignées de zéro sur la ligne de temps. L'impact sur l'ordonnancement est qu'il y a plus de possibilité d'ordonnancement, car la première fenêtre d'échéance est loin de zéro. Au contraire, un facteur R de 0,4 donne des bornes plus près et même collées sur le zéro. L'impact sur l'ordonnancement est que le nombre de possibilités diminue, car il est possible d'ordonnancer moins de tâches avant la première fenêtre. Alors les ordonnancements se ressemblent d'une méthode à une autre, d'où le regroupement sur les instances 10, 11 et 12.

4.4 Temps d'exécution

Le facteur temps est très important en industrie. L'évaluation des ensembles PO ne se base pas exclusivement sur la qualité et la distribution des solutions mais aussi sur le temps d'exécution des méthodes.

Le Tableau 19 présente le temps d'exécution moyen de l'ensemble des méthodes adaptées dans ce mémoire.

	NSGAII	PMSMO	GISMOO	ACISMOO	ACISMOOmodif	BCMC
20	4,95	6,37	12,42	4,95	4,93	27,22
30	9,83	11,95	20,70	10,55	9,28	67,05
40	18,35	21,33	32,97	19,02	16,05	143,72
50	30,15	32,97	50,02	28,98	25,30	266,53
75	98,57	108,77	156,27	87,00	73,57	904,15
100	219,03	251,03	355,95	184,82	152,18	1964,98

Tableau 19 : Temps d'exécution moyen (en secondes)

Les lignes représentent le nombre de tâches des groupes d'instances et les colonnes, les méthodes étudiées. La jonction d'une ligne et d'une colonne représente la moyenne du temps des cinq exécutions des groupes d'instances. ACISMOOmodif a le temps d'exécution minimal et BCMC, le maximal. Le temps d'exécution des méthodes d'OCF uni-colonie est très près du temps d'exécution du NSGA-II. La différence de temps entre les méthodes d'OCF uni-colonie est négligeable. GISMOO est généralement deux fois plus lent que les méthodes d'OCF uni-colonie. Ceci s'explique par le fait que la population de GISMOO est de 100 individus et la colonie d'ACISMOO est de 20 fourmis, tel que recommandé par l'ACS uni-objectif. Le temps d'exécution de BCMC est cinq fois supérieurs à l'ACISMOO sur les instances de 20 tâches pour le même critère d'arrêt, soit 100 000 évaluations. Cette différence est essentiellement causée par le fait que BCMC a une base AS, donc sélectionne une tâche en utilisant une méthode probabiliste, ce qui augmente le temps d'exécution. La différence entre les autres instances augmente de façon exponentielle, causée par le nombre total de fourmis de BCMC, qui est égal au nombre de tâches, et que pour ACISMOO le nombre de fourmis demeure constant à 20. Le nombre total de fourmis d'ACISMOO et ACISMOOmodif est fixé à 20 pour conserver les

caractéristiques des OCF uni-objectif. Également, la diversité d'ACISMOO et ACISMOOmodif ne vient pas du nombre total de fourmis, mais des paramètres dynamiques.

CHAPITRE 5

CONCLUSION

Les entreprises de tous les secteurs sont confrontées à des PMO, car elles doivent résoudre des problèmes complexes, de grandes dimensions et généralement, les objectifs sont contradictoires ainsi que non-commensurables. La majorité des PMO, dont les problèmes de machine unique, sont réputés NP-difficiles, ce qui en font des problèmes intéressants académiquement (Garey et al. 1979; Wan et al. 2013). Pour se rapprocher des contextes réels, les problèmes de machine unique prennent en considération différentes dimensions afin d'être représentatifs de la pratique. Le problème de machine unique multi-objectifs présenté par Arroyo et al (2011) (MURMO) a plusieurs caractéristiques se rapprochant de contextes réels. Ces caractéristiques sont le traitement multi-objectifs du problème, les réglages dépendants de la séquence, les fenêtres d'échéance pour chaque tâche et la possibilité d'insérer des temps morts dans l'ordonnancement. Le mémoire répond au premier objectif secondaire, soit de *résoudre efficacement un PMO ayant des aspects traitant de contextes réels,* en adaptant des méthodes de résolution au problème MURMO. L'efficacité des méthodes de résolution est évaluée à l'aide des métriques. La Section 4.2 présente les résultats. Généralement, ACISMOOmodif génère des solutions de meilleure qualité qu'ACISMOO et BCMC. ACISMOOmodif a une meilleure répartition le long de le front Pareto pour les instances de 20 et 30 tâches, selon la métrique Espacement. Toutefois, la représentation graphique d'une instance typique a démontré qu'ACISMOOmodif diversifie mieux sa recherche qu'ACISMOO. Également, toutes les approches d'OCF de ce mémoire génèrent des solutions de qualité inférieure à GISMOO.

La comparaison entre les méthodes d'évaluation occupe une part importante de ce mémoire. Elle représente le second objectif secondaire qui est d'effectuer la *comparaison*

équitable entre une méthode uni-colonie et une méthode multi-colonies. Cet objectif est atteint avec l'adaptation des méthodes BCMC et ACISMOO à la Section 3.4. La comparaison peut être qualifiée d'équitable, car les conditions d'expérimentation ont été judicieusement choisies et appliquées (Section 4.1). Les conclusions de la comparaison sont qu'ACISMOO génère des solutions de meilleure qualité que BCMC. Généralement, ACISMOO est mieux réparti le long de le front Pareto.

Une des différences entre BCMC et ACISMOO est l'utilisation de concepts AE Pareto. La transposition de GISMOO vers un ACS qui crée ACISMOO permet de répondre au troisième objectif qui est d'offrir *la transposition de concepts AE Pareto dans une approche d'OCF*. Cette transposition permet d'améliorer la diversité de la méthode par la création de 50% de la colonie avec un système immunitaire artificiel. Les concepts AE Pareto sont également utilisés dans la mise-à-jour des phéromones. La mise-à-jour est alors réalisée par un plus petit nombre de fourmis que le BCMC, ce qui favorise ACISMOO en termes de temps d'exécution.

Une autre différence entre BCMC et ACISMOO est le nombre de colonies. La littérature a démontré que de spécialiser les fourmis dans une méthode uni-colonie est avantageux (Angus 2007). La spécialisation d'une fourmi est le principe où les paramètres dynamiques (seuil de la règle de transition (q_0), les exposants liés à l'importance de la visibilité $(\theta \; et \; \varpi)$) d'une fourmi sont différents d'une autre. Ce principe n'a pas été exploité à son plein potentiel avec ACISMOO. C'est pour cette raison que le dernier objectif secondaire *suggère une amélioration performante de la méthode ACISMOO*. La comparaison entre ACISMOO et ACISMOOmodif permet d'observer que ce dernier trouve

des solutions de meilleure qualité. Les deux méthodes sont équivalentes en termes d'uniformité de la répartition des solutions. Toutefois, seules les instances de 20 tâches présentent une bonne répartition.

C'est au travers des objectifs secondaires que ce mémoire répond à son objectif principal, qui est de *proposer des approches de résolutions multi-objectifs performantes afin de permettre un rapprochement entre la théorie et la pratique.* Les objectifs secondaires regroupés répondent à cet objectif en permettant de résoudre efficacement un problème avec des concepts se rapprochant des contextes réels (MURMO) et en proposant six méthodes de résolutions qui se basent sur différents algorithmes : AE et OCF.

Cette recherche comprend quelques limitations. Lors de la comparaison entre GISMOO et ACISMOO, la conclusion est que GISMOO génère un front Pareto plus continu qu'ACISMOO. Toutefois ACISMOO génère des solutions non-dominées extrêmes que GISMOO ne découvre pas. Également, ACISMOO n'est adapté que pour le problème MURMO. Aussi, ACISMOO est difficilement généralisable à plus de deux objectifs pour le moment, ceci est essentiellement causé par le problème choisi. Cette constatation est essentiellement causée par l'utilisation de la méthode OTA pour calculer la date de fin des tâches.

La recherche offre de multiples ouvertures. La première ouverture est de faire plus ample recherche pour diversifier ACISMOO en conservant une bonne répartition des solutions sur le front Pareto. Dit autrement, diversifier ACISMOO sans que son front Pareto ne soit disjoint. Cette adaptation d'ACISMOO permet également de découvrir si la frontière disjointe est causée par le problème MURMO ou la méthode. Deuxièmement, en

adaptant ACISMOO pour d'autres problèmes de la littérature, cela amène une image plus globale du fonctionnement de la méthode. Une généralisation à plusieurs objectifs des méthodes d'OCF proposées est une autre extension de la recherche possible. Le but est de permettre d'utiliser ACISMOO sur des problèmes à plus de deux objectifs, ce qui est courant en industrie. Également, les mutations utilisées pour ACISMOO sont les mêmes que MONVNS3 (Arroyo et al. 2011) dans le but de rendre la comparaison équitable. Dans l'optique d'améliorer les performances d'ACISMOO, une extension de la recherche consiste à modifier les mutations pour considérer, par exemple, le réglage dépendant de la séquence avec une mutation de type bloc. Une nouvelle extension de la recherche est de tester le problème MURMO sur d'autres formes d'hybridation. Étant donné que l'hybridation de deux métaheuristiques donne de bons résultats, tel que le démontrent GISMOO et ACISMOO. Une autre possibilité est la transposition des concepts AE Pareto vers un autre ACS Pareto qui ne les utilise pas. Le mémoire établit la force de l'utilisation des concepts AE Pareto. Intégrer ceux-ci dans un ACS Pareto, par exemple BCMC, qui n'en n'a pas permet de voir la porté de ces concepts. Pour finir, différentes approches peuvent être étudiées pour évaluer la dominance. La littérature en compte déjà plusieurs : *a-dominance* (Othmani 1998), *Geoffrion* (Ehrgott 2005), *cône-dominance* (Collette et al. 2002), sans être exhaustif. L'approche d'évaluation de la dominance d'une solution a un fort impact sur le front Pareto généré. Quel est l'impact des différentes approches d'évaluation? Il y a-t-il des approches qui tendent vers la convergence ou la diversification des solutions? Aussi, pourquoi ne pas réaliser une hybridation où plusieurs approches d'évaluation de la dominance sont utilisées?

BIBLIOGRAPHIE

Allahverdi, A., Ng, C. T., Cheng, T. C. E., and Kovalyov, M. Y. 2008. "A survey of scheduling problems with setup times or costs," *European Journal of Operational Research* (187:3) 6/16/, pp 985-1032.

Angus, D. 2007. "Crowding Population-based Ant Colony Optimization for the Multi-objective Travelling Salesman Problem," in *Proceedings of the 2007 IEE Symposium on Computational Intelligence in Multicriteria Decision Making (MCDM'2007)*: Honolulu, Hawaii USA pp. 333-340.

Angus, D., and Clinton, W. 2009. "Multiple objective ant colony optimisation," *SWARM INTELLIGENCE* (3:1).

Arroyo, J. E. C., Ottoni, R. d. S., and Oliveira, A. d. P. 2011. "Multi-objective Variable Neighborhood Search Algorithms for a Single Machine Scheduling Problem with Distinct due Windows," *Electronic Notes in Theoretical Computer Science* (281), pp 5-19.

Beckers, R., Deneubourg, J. L., and Goss, S. 1992. "Trails and U-Turns in the Selection of a Path by the Ant Lasius-Niger," *Journal of Theoretical Biology* (159:4) Dec 21, pp 397-415.

Berrichi, A., Yalaoui, F., Amodeo, L., and Mezghiche, M. 2010. "Bi-Objective Ant Colony Optimization approach to optimize production and maintenance scheduling," *Computers & Operations Research* (37:9), pp 1584-1596.

Berro, A. 2001. *Optimisation multiobjectif et stratégies d'évolution en environnement dynamique*, Université des Sciences Sociales Toulouse I, Toulouse.

Biskup, D., and Jahnke, H. 2001. "Common due date assignment for scheduling on a single machine with jointly reducible processing times," *International Journal of Production Economics* (69:3), pp 317-322.

Bullnheimer, B., Hartl, R. F., and Strauss, C. 1997. "A new rank based version of the Ant System - A computational study.," *Technical Report, Institute of Management Science, University of Vienna, Austria*).

Chaharsooghi, S. K., and Meimand Kermani, A. H. 2008. "An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP)," *Applied Mathematics and Computation* (200:1), pp 167-177.

Charnes, A., and Cooper, W. W. 1961. *Management models and industrial applications of linear programming. Vol. II*, (John Wiley \& Sons Inc.: New York.

Collette, Y., and Siarry, P. 2002. *Optimisation multiobjectif*, (Ed. Techniques Ingénieur.

Colorni, A., Dorigo, M., and Maniezzo, V. 1992. "DISTRIBUTED OPTIMIZATION BY ANT COLONIES," in *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, F. J. Varela and P. Bourgine (eds.), M I T Press: Cambridge, pp. 134-142.

Conner, G. 2009. "10 Questions," *Manufacturing Engineering* (142:3) 20090301, pp 93-96,98-99.

Corne, D., Jerram, N. R., Knowles, J. D., and Oates, M. J. 2001. "PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference GECCO'2001*, pp. 283--290.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. 2000. "A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimisation: NSGA-II," in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, Springer-Verlag, pp. 849-858.

Deb, K., Pratap, A., Agarwal, S., and Mayarivan, T. 2002. "A Fast and Elitist Multi-objective Genetic Algorithm : NSGA-II " *IEE Transaction on Evolutionary computation* (6:2), pp 182-197.

Dhaenens-Flipo, C. 2005. *Optimisation combinatoire multi-objectif: apport des méthodes coopératives et contribution à l'extraction de connaissances*, Université des sciences et technologies de Lille, Lille, France.

Doerner, K., Hartl, R. F., and Reimann, M. 2003. "Are COMPETants more competent for problem solving? - the case of a multiple objective transportation problem," *Central European Journal of Operations Research* (11:2), pp 115-141.

Dorigo, M., and Gambardella, L. M. 1997. "Ant Colonies for the Traveling Salesman Problem.," *BioSystems* (43), pp 73-81.

Dorigo, m., Maniezzo, V., and Colorni, A. 1996. "The Ant System : Optimization by a Colony of Cooperating Agents.," *IEEE Transactions on Systems, Man, and Cybernetics–Part B* (26), pp 29-41.

Dorigo, M., and Stützle, T. 2004. *Ant colony optimization*, (MIT Press: Cambridge, Mass.

Dréo, J., and Siarry, P. 2003. *Métaheuristiques pour l'optimisation difficile*, (Eyrolles: Paris.

Du, J., and Leung, J. Y. T. 1990. "MINIMIZING TOTAL TARDINESS ON ONE MACHINE IS NP-HARD," *Mathematics of Operations Research* (15:3), pp 483-495.

Edgeworth, F. Y. 1881. *Mathematical Psychics: An essay on the application of mathematics to the moral sciences.*, (

Ehrgott, M. 2005. *Multicriteria optimization*, (Springer.

Francisci, D. 2002. "Algorithmes évolutionnaires et optimisation multi-objectifs en data mining," Laboratoire informatique, signaux et systèmes de Sophia Antipolis UMR 6070, Sophia-Antipolis Cedex, France.

Fujita, K., Hirokawa, S., Akagi, S., Kimatura, S., and Yokohata, H. 1998. "Multi-objective optimal design of automotive engine unisg genetic algorithm," in *Design Engineering Tehcnical Conferences DETC'98*: Atlanta, Georgia, pp. 1-11.

Gagné, C., LeBel, A., Zinflou, A., and Yalaoui, F. Year. "Étude comparative d'approches Pareto pour un problème d'ordonnancement multi-objectifs avec réglages et retard," 14e conférence ROADEF de la Société Française de Recherche Opérationnelle et Aide à la Décision, Université de Techonologie de Troyes, France, 2013.

Gambardella, L. M., and Dorigo, M. Year. "Ant-Q: A Reinforcement Learning approach to the traveling salesman problem," Proceedings of the Twelfth International conference on Machine Learning, Palo Alto CA, 1995, pp. 252-260.

Gambardella, L. M., Taillard, E. D., and Giovanni, A. 1999 "MACS-VRPTW : A multiple ant colony system for vehicule routing problems with ant colony system for vehicle routing problems with th windows," in *New ideas in optimization*, M. Dorigo and F. Glover (eds.): McGraw-Hil, pp. 73-76.

Garcia-Martinez, C., Cordon, O., and Herrera, F. 2007. "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," *European Journal of Operational Research* (180:1) Jul, pp 116-148.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, (W. H. Freeman& Co.

Goldberg, D. E. 1989. *Genetic algorithms in search, optimization, and machine learning*, (Addison-Wesley: Reading, Mass.

Gordon, V., Proth, J.-M., and Chu, C. 2002. "A survey of the state-of-the-art of common due date assignment and scheduling research," *European Journal of Operational Research* (139:1) 5/16/, pp 1-25.

Goss, S., Aron, S., Deneubourg, J. L., and Pasteels, J. M. 1989. "Self-organized shortcuts in the Argentine ant," *Naturwissenschaften* (76:12) 1989/12/01, pp 579-581.

Govindarajan, L. K. T. 2005. "MULTIOBJECTIVE OPTIMIZATION OF PROCESS PLANT USING GENETIC ALGORITHM," *International Journal of Computational Intelligence & Applications* (5:4), pp 425-437.

Gravel, M., Gagné, C., and Wilson, L. P. 2002. "Scheduling continuous casting of aluminium using a multiple objective ant colony optimization metaheuristic," *European Journal of Operational Research* (143:1), pp 218-229.

Graves, G. H., and Lee, C. Y. 1999. "Scheduling maintenance and semiresumable jobs on a single machine," *Naval Research Logistics (NRL)* (46:7), pp 845-863.

Grosan, C. 2003. "Performance metrics for multiobjective optimization evolutionary algorithms,").

Guntsch, M., and Middendorf, M. 2003. "Solving Multi-Criteria Optimization Problems with Population-Based ACO," in *Evolutionary Multi-Criteria Optimization (EMO'03)*: Faro, Portugal, pp. 464-478.

Hall, N. G., Kubiak, W., and Sethi, S. P. 1991. "Earliness–tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date," *Operations Research* (39:5), pp 847-856.

Hwang, C. L., and Masud, A. S. M. 1980. *Multiple objective decision making, methods and applications: a state-of-the-art survey*, (Springer-Verlag.

Iredi, S., Merkle, D., and Middendorf, M. 2001. "Bi-criterion optimization with multi colony ant algorithms," in *Evolutionary Multi-Criterion Optimization, Proceedings*, E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello and D. Corne (eds.), Springer-Verlag Berlin: Berlin, pp. 359-372.

Kumar, R. P. 2002. "Improved Sampling of the Pareto-Front in Multiobjective Genetic Optimizations by Steady-State Evolution: A Pareto Converging Genetic Algorithm," *Evolutionary Computation* (10:3) Fall2002, pp 283-314.

Laumanns, M., Zitzler, E., and Thiele, L. Year. "A unified model for multi-objective evolutionary algorithms with elitism," Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, IEEE2000, pp. 46-53.

Li, L. H. G. Q. 2009. "Multiobjective evolutionary optimisation for adaptive product family design," *International Journal of Computer Integrated Manufacturing* (22:4), pp 299-314.

Liu, N., Hang, B., and Pan, X. H. 2005. "Using the Ant algorithm to Derive Pareto Fronts for Multiobjective siting of Emergency service facilities," *Transportation*

Research Record : Journal of the Transportation Research board (1935), pp 120-129.

Liu, Z., and Cheng, T. 2002. "Scheduling with job release dates, delivery times and preemption penalties," *Information Processing Letters* (82:2), pp 107-111.

López-Ibáñez, M., Paquete, L., and Stützle, T. 2004. "On the Design of ACO for the Biobjective Quadratic Assignment Problem Ant Colony Optimization and Swarm Intelligence," M. Dorigo, M. Birattari, C. Blum, L. Gambardella, F. Mondada and T. Stützle (eds.), Springer Berlin / Heidelberg, pp. 436-459.

Maniezzo, V. 1999. "Exact and Approximate Nondeterministic Tree-Search Procedures for the Quadratic Assignment Problem," *INFORMS J. on Computing* (11:4), pp 358-369.

Mariano, C. E., and Morales, E. 1999. "A multiple objective Ant-Q algorithm for the disign of water distribution irrigation networks," HC-9904, Instituto Mexicano de Tecnologia del Agua, Mexico.

Merkle, D., and Middendorf, M. 2000. "An Ant Algorithm with a New Pheromone Evaluation Rule for Total Tardiness Problems," in *Real-World Applications of Evolutionary Computing*, S. Cagnoni (ed.), Springer Berlin Heidelberg, pp. 290-299.

Moncayo-Martinez, L. A., and Zhang, D. Z. 2011. "Multi-objective ant colony optimisation: A meta-heuristic approach to supply chain design," *International Journal of Production Economics* (131:1) May, pp 407-420.

Obayashi, S., Takahashi, S., and Takeguchi, Y. 1998. "Niching and Elitist Models for MOGAs," in *5th International Conference Parallel Problem Solving from Nature PPSN'5,* Springer-Verkag (ed.): Amsterdam, pp. 260-269.

Oliveira, L. S. S. R. B. F. S. C. Y. 2003. "A Methodology for Feature Selection Using Multiobjective Genetic Algorithms for Handwritten Digit String Recognition," *International Journal of Pattern Recognition & Artificial Intelligence* (17:6), p 903.

Othmani, I. 1998. *Optimisation multicritère: fondements et concepts*, Université Joseph-Fourier-Grenoble I.

Panwalkar, S., Dudek, R., and Smith, M. Year. "Sequencing research and the industrial scheduling problem," Symposium on the Theory of Scheduling and its Applications, Springer1973, pp. 29-38.

Pareto, V. 1896. *Cours d'Économie Politique.*, (Rouge, Lausanne, Switzerland.

Pinedo, M. 2012. *Scheduling : theory, algorithms, and systems*, (4nd ed.) Prentice-Hall: Upper Saddle River, N.J.

Sayin, S., and Karabati, S. 1999. "A bicriteria approach to the two-machine flow shop scheduling problem," *European Journal of Operational Research* (113:2) Mar, pp 435-449.

Schaffer, J. D. 1985. "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, L. Erlbaum Associates Inc., pp. 93-100.

Schott, J. R. 1995. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization.*, Master, AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH.

Sioud, A., Gravel, M., and Gagné, C. Year. "New crossover operator for the single machine scheduling problem with sequence-dependent setup time," 2009 International Conference on Genetic and Evolutionary methods. GEM09, Las Vegas, 2009, pp. 79-84.

Stützle, T., and Hoos, H. H. 2000. "MAX–MIN Ant System," *Future Generation Computer Systems* (16:8) 6//, pp 889-914.

Talbi, E.-G. 1999. "Métaheuristiques pour l'optimisation combinatoire multi-objectif : État de l'art," *CNET*), p 34.

Talbi, E.-G. 2009. *Metaheuristics : from design to implementation*, (John Wiley & Sons: Hoboken, N.J.

Tamura, H. 2007. "A new multiobjective genetic algorithm with heterogeneous population for solving flowshop scheduling problems," *International Journal of Computer Integrated Manufacturing* (20:5), pp 465-477.

Tan, K., and Narasimhan, R. 1997. "Minimizing tardiness on a single processor with sequence-dependent setup times: a simulated annealing approach," *Omega* (25:6), pp 619-634.

Teixeira, C., Covas, J. A., Stutzle, T., and Gaspar-Cunha, A. 2012. "Multi-objective ant colony optimization for the twin-screw configuration problem," *Engineering Optimization* (44:3), pp 351-371.

Van Veldhuizen, D. A. 1999. "Multiobjective evolutionary algorithms: classifications, analyses, and new innovations," DTIC Document.

Wan, G., and Yen, B. P. C. 2002. "Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties," *European Journal of Operational Research* (142:2), pp 271-281.

Wan, L., and Yuan, J. 2013. "Single-machine scheduling to minimize the total earliness and tardiness is strongly NP-hard," *Operations Research Letters* (41:4) 7//, pp 363-365.

Wisner, J. D., and Siferd, S. P. 1995. "A Survey of U.S. Manufacturing Practices in Make-To-Order Machine Shops," *Production and Inventory Management Journal* (36:1), pp 1-7.

Yang, Y., Wu, G., Chen, J., and Dai, W. 2010. "Multi-objective optimization based on ant colony optimization in grid over optical burst switching networks," *Expert Systems with Applications* (37:2), pp 1769-1775.

Zhou, G. G., and Gen, M. 1999. "Genetic algorithm approach on multi-criteria minimum spanning tree problem," *European Journal of Operational Research* (114:1) Apr, pp 141-152.

Zhu, X., and Wilhelm, W. E. 2006. "Scheduling and lot sizing with sequence-dependent setup: A literature review," *IIE transactions* (38:11), pp 987-1007.

Zinflou, A. 2008. *Algorithmes évolutionnaires pour l'ordonnancement industriel : application à l'industrie automobile*, Université du Québec à Chicoutimi, Chicoutimi.

Zinflou, A., Gagné, C., and Gravel, M. 2012. "GISMOO: A new hybrid genetic/immune strategy for multiple-objective optimization," *Computers & Operations Research* (39:9), pp 1951-1968.

Zinflou, A., Gagné, C., Gravel, M., and Price, W. 2008. "Pareto memetic algorithm for multiple objective optimization with an industrial application," *Journal of Heuristics* (14:4), pp 313-333.

Zitzler, E. 1999. *Evolutionaty algorithms for multiobjective optimization : Methods and applications*, Siss Federal Institute of Technology Zurich.

Zitzler, E., Laumanns, M., and Thiele, L. 2001. "SPEA2 : Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN 2001. Evolutionaty Methods for Design, Optimisation and Control with Applications to industrial Problems*: Athens, Greece, pp. 95-100.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. G. 2003. "Performance assessment of multiobjective optimizers: An analysis and review," *Ieee Transactions on Evolutionary Computation* (7:2) Apr, pp 117-132.