UQAC
Université du Québec
à Chicoutimi

Amélioration des stratégies métaheuristiques à base de population pour la résolution de problèmes de grande taille

par Hugo Deschênes

Thèse présenté à l'Université du Québec à Chicoutimi en vue de l'obtention du grade de PhD au doctorat en Sciences et technologies de l'information, en extension de l'Université du Québec en Outaouais

Québec, Canada

RÉSUMÉ

L'industrie d'aujourd'hui est constamment en quête de nouvelles méthodes pour améliorer ses processus et ses activités. Elle a recours à la recherche opérationnelle pour répondre à ses besoins. Les problèmes auxquels elle fait face peuvent être définis par l'utilisation de variables de décision entières (problèmes de nature discrète) ou de variables de décision réelles (problèmes de nature continue ou mixte). Les métaheuristiques permettent d'approcher ces problèmes lorsqu'ils sont NP-difficiles. Cette catégorie de méthodes offre un moyen de trouver une solution acceptable dans un délai raisonnable. L'objectif général de cette thèse consiste à améliorer les stratégies métaheuristiques pour la résolution de problèmes de grande taille. Parmi les métaheuristiques bien connues, la famille des algorithmes d'optimisation par essaims particulaires (PSO) offre d'excellentes performances pour la résolution de problèmes de nature continue.

Le PSO a beaucoup évolué depuis sa conception. Deux stratégies sont utilisées afin d'améliorer ses performances : l'hybridation et la discrétisation. L'hybridation permet d'améliorer les performances en combinant les forces des méthodes d'optimisation utilisées. La discrétisation permet d'étendre l'applicabilité d'une méthode d'optimisation à une plus grande variété de problèmes. Il est bien connu que plus la dimension d'un problème augmente, plus il s'avère difficile de résoudre ce problème efficacement. Malgré les forces du PSO, cette métaheuristique éprouve de la difficulté à résoudre les problèmes de grande taille.

Cette thèse propose tout d'abord la conception de trois algorithmes hybrides construits selon trois variantes de PSO : le *Barebones* PSO, le *Comprehensive Learning* PSO et le *Cooperative Learning* PSO. L'utilisation de fonctions continues bien connues (Rosenbrock, Ackley, etc.) et de fonctions provenant du *CEC'15 benchmark set* permettent de statuer sur les modèles hybrides les plus prometteurs permettant de pallier les lacunes du PSO à résoudre des instances de grande taille.

Suivant cette idée, nous proposons par la suite différents processus de discrétisation pour la résolution de problèmes de support à complexité variable : les problèmes d'ordonnancement séquentiel et d'ordonnancement de projets avec contraintes de ressources. Les conclusions obtenues sur la performance des processus de discrétisation permettent d'amener le PSO à résoudre plus efficacement des problèmes complexes de nature entière.

Les contributions précédentes sont réutilisées afin de valider l'extensibilité du PSO. Le problème d'alignement multiple de séquences (MSA) est un problème combinatoire bien connu offrant un grand niveau de complexité et des instances de très grande taille. Il est utilisé pour éprouver les améliorations apportées au PSO. Une version hybride et discrète du PSO est finalement proposée afin de résoudre le MSA. Pour ce faire, une codification de solutions sous forme d'anneau et spécialisée en fonction du MSA est définie. Une amélioration est ensuite proposée définissant un nouveau comportement pour le PSO permettant de reproduire le phénomène de magnétisme dans le but d'aider l'algorithme à générer de bonnes solutions.

Les algorithmes proposés dans le cadre de cette thèse démontrent plusieurs stratégies permettant d'améliorer le PSO pour la résolution de grandes instances de problèmes. Ces stratégies peuvent être appliquées à davantage de métaheuristiques continues à base de population et permettent de contribuer à leur flexibilité. Elles peuvent également être appliquées pour résoudre une plus grande variété de problèmes discrets à complexité variable.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES

ADN	Acide désoxyribonucléique
GPSO	Global PSO
LPSO	Local PSO
LT-Index	Liste de Translations – Index
LT-Proportion	Liste de Translations – Proportion
MH	Métaheuristique
MSA	Alignement multiple de séquences (*Multiple sequence alignement*)
OP	Opérateur arithmétique
PI	Pseudo-insertion
PSO	Optimisation par essaims particulaires (*Particle swarm optimization*)
RCPSP	Ordonnancement de projets avec contraintes de ressources (*Resource Constrained Project Scheduling Problem*)
SOP	Ordonnancement séquentiel
SP	Somme des pairs de séquences

LISTE DES ABRÉVIATIONS

a	Variable de décision	A	Ensemble des variables des décision
$\boldsymbol{b}$	Vecteur réel		
c	Paramètre de confiance du PSO		
d	Numéro d'une dimension	D	Nombre total de dimensions
e	Numéro d'un espacement	E	Nombre total d'espacements
f	Fonction objectif		
g	Meilleure particule du PSO		
h	Une métaheuristique		
i	Numéro d'une particule		
j	Numéro d'une séquence du MSA	J	Nombre total d'activités du RCPSP
k	Numéro d'une séquence du MSA	K	Nombre total de villes du SOP
m	Compteur du CLPSO	M	Valeur critique du compteur du CLPSO
		N	Nombre total de particules
p	Meilleure solution cognitive du PSO	P	Procédé de discrétisation
		Pc	Probabilité d'apprentissage
q	Vecteur solution		
r	Nombre aléatoire	R	Nombre de ressources différentes pour le RCPSP
s	Séquence du MSA	S	Nombre total de séquences
t	Numéro de l'itération d'un algorithme	T	Nombre total d'itérations
u	Numéro d'une permutation	U	Nombre total de permutations
v	Vélocité du PSO		
w	Paramètre d'inertie du PSO	W	Facteur de constriction du PSO
x	Position du PSO		
y	Numéro d'une pseudo-insertion	Y	Nombre total de pseudo-insertions
z	Longueur d'un espacement		

α	Pointage d'un alignement du MSA		
β	Pointage d'un espacement du MSA		
δ	Pénalité pour l'agrandissement d'un espacement du MSA		
ε	Pénalité pour l'ouverture d'un espacement du MSA		
γ	Numéro de l'itération du magnétisme	Γ	Nombre maximal d'appels au magnétisme
μ	Moyenne pour une distribution normale		
σ	Variance pour une distribution normale		
λ	Valeur du déplacement d'une pseudo-insertion		
ρ	Accélération du calcul haute performance		
φ	Efficacité du calcul haute performance		

DÉDICACE

À ma très chère amie Annie Lebel, sans qui rien de tout ça n'aurait de sens,

À mes nièces qui évoluent dans la société de demain,
À ma sœur et à mes parents qui m'ont tout appris,
À mon amour qui me soutient inconditionnellement,
À mes ami(e)s qui m'ont vu tomber et me relever à multiples reprises,
À ma deuxième famille au crossfit sans qui j'aurais perdu toute ma tête,
Aux cobras, *always*.

REMERCIEMENTS

J’aimerais tout d’abord remercier ma directrice de recherche, Mme Caroline Gagné. Cette femme incroyable m’a enseigné bien plus que l’art de construire un doctorat et d’effectuer de la recherche. Son support, ses valeurs et son sens critique m’auront été inculqués et je n’aurais souhaité d’une meilleure personne pour m’accompagner dans ce grand projet. Merci beaucoup pour tout, du profond de mon cœur.

Merci aux membres de mon jury d’évaluation pour tout le travail essentiel que vous faites pour mon accomplissement académique. Plus précisément, mes remerciements vont à Mme Caroline Gagné, M. Alexandre Blondin-Massé, M. Michael Krajecki et à la présidence de M. Hamid Mcheick sur ce comité.

J’aimerais également remercier le CRSNG pour avoir financé une partie de mes recherches pendant mon parcours académique. Leur aide financière m’a été fortement utile.

Finalement, merci à l’UQAC, l’UQO et au département d’informatique et de mathématiques de l’UQAC pour m’avoir permis de compléter ce doctorat sous leur bannière. Ce fût un honneur!

AVANT-PROPOS

J'étais au baccalauréat à l'UQAC lorsque j'ai découvert le domaine de la recherche opérationnelle, dans un cours de 1er cycle avec Mme Caroline Gagné. Le croisement des mathématiques et de l'informatique m'a toujours fasciné, où la résolution de problèmes complexes demande des compétences en programmation et un raisonnement logique. Lors de mes études de 2e cycle universitaire, j'ai appris et maîtrisé la structure et le comportement de l'optimisation par essaims particulaires pour la résolution du problème d'ordonnancement de projets avec contraintes de ressources. Le présent ouvrage poursuit les recherches effectuées en poussant plus loin les stratégies de conception des métaheuristiques pour la résolution de problèmes d'une plus grande complexité, mais surtout, pour les problèmes de grande taille. L'algorithme d'optimisation par essaims particulaires est considéré avoir de la difficulté pour la résolution de grandes instances de problèmes. Les recherches dans ce document et les contributions effectuées visent à étendre l'applicabilité du PSO et à proposer des stratégies lui permettant d'être davantage efficace pour la résolution d'une plus grande variété de problèmes. Cette thèse aborde des sujets tels que la résolution de fonctions continues et de problèmes discrets, les métaheuristiques, la discrétisation, l'hybridation, de même que la bio-informatique avec la résolution du problème d'alignement multiple de séquences.

CHAPITRE 1 – INTRODUCTION GÉNÉRALE

1.1. BASES THÉORIQUES

Cette thèse porte sur certaines stratégies métaheuristiques en recherche opérationnelle. Elle est réalisée dans le but de contribuer à la résolution de problèmes de grande taille par l'apport de nouveaux comportements permettant d'améliorer la qualité des solutions. Pour y parvenir, une revue de la littérature est effectuée sur des sujets établissant les bases des contributions et des expérimentations. Ces contributions concernent l'hybridation, la discrétisation, l'optimisation par essaims particulaires (PSO) et la résolution d'instances de grande taille. Elles seront validées par la résolution de fonctions continues, des problèmes d'ordonnancement séquentiel (SOP), d'ordonnancement de projets avec contraintes de ressources (RCPSP) et d'alignement multiple de séquences (MSA).

La mise en contexte du sujet débute avec la définition des problèmes à résoudre. Un problème d'optimisation est en effet défini selon trois éléments, soient l'espace de recherche, une ou plusieurs fonctions objectifs, de même qu'un ensemble de contraintes à respecter (Ravindran, 2016). Boussaïd et al. (2013) mentionnent que la nature des variables de décision qui composent l'espace de recherche définit le type de problème dont il est question : un problème continu (variables réelles), un problème discret (variables entières) ou un problème mixte (variables réelles et entières). Plusieurs de ces problèmes font partie d'une catégorie dite NP-difficiles, lesquels favorisent l'utilisation d'une méthode approchée de résolution : les métaheuristiques. Celles-ci peuvent être particulièrement efficaces pour la résolution de ce type de problèmes si elles sont bien adaptées au problème à résoudre.

Les métaheuristiques utilisent des stratégies de conception permettant d'équilibrer l'exploration et l'exploitation de l'espace de recherche. Cet ajustement permet de concevoir une métaheuristique performante selon un problème particulier (El-Ghazali Talbi, 2009). On distingue deux grandes catégories de métaheuristiques : à solution unique et à base de population (Boussaïd et al., 2013). Cette dernière catégorie se divise en deux sous-

catégories, soient les algorithmes évolutionnaires et les algorithmes d'intelligence en essaims. Le PSO est une métaheuristique faisant partie de cette deuxième sous-catégorie (R. Eberhart & Kennedy, 1995; Y. Shi & Eberhart, 1998). Les algorithmes d'intelligence en essaims sont d'ailleurs reconnus pour être flexibles et robustes (Lalwani et al., 2015).

Deux stratégies de conception sont étudiées dans le cadre de cette thèse : l'hybridation et la discrétisation. La première stratégie, l'hybridation, implique la combinaison d'au moins deux méthodes d'optimisation en un seul processus. Une variété d'hybridations sont exposées dans la littérature (Banks et al., 2008; Raidl, 2015). E-G Talbi (2002) établit certaines caractéristiques hiérarchiques permettant de les catégoriser. Parmi celles-ci, la nature homogène et hétérogène d'un hybride est abordée. Un hybride homogène fait intervenir deux métaheuristiques de nature identique et un hybride hétérogène fait intervenir deux métaheuristiques de nature différente.

La deuxième stratégie, la discrétisation, implique les moyens utilisés afin d'adapter une métaheuristique continue pour la résolution d'un problème composé de variables de décisions binaires ou entières. Il est connu que le nombre de métaheuristiques conçues pour la résolution de problèmes continus est plus grand que celui traitant des variables entières (Fister Jr et al., 2013; Hussain et al., 2019). Une partie de la littérature cherche à adapter ces métaheuristiques afin de les rendre plus polyvalentes. Krause et al. (2013) mettent de l'avant six procédés de discrétisation permettant d'adapter les métaheuristiques continues d'intelligence en essaims pour la résolution de problèmes discrets. Le sixième procédé est réservé pour les adaptations impliquant le PSO.

Les stratégies utilisées dans cette thèse sont appliquées à une métaheuristique d'intelligence en essaims bien connue : le PSO (R. Eberhart & Kennedy, 1995; Y. Shi & Eberhart, 1998). Cet algorithme est reconnu pour sa malléabilité, ses bonnes performances et sa facilité d'adaptation pour la résolution de problèmes continus (Banks et al., 2008). Selon

Krause et al. (2013), plusieurs des algorithmes continus à base de population ayant servi pour la résolution de problèmes combinatoires utilisent cette métaheuristique. La littérature contient de nombreux exemples d'hybridations réussies (Hussain et al., 2019) et de procédés de discrétisation utilisant cette métaheuristique (Anghinolfi & Paolucci, 2009; Clerc, 2004; H. Zhang et al., 2005). Les défis de l'adaptation de ces stratégies de conception par l'utilisation du PSO sont constants et nécessitent davantage de recherche afin de poursuivre les améliorations. Les recherches sont d'autant plus pertinentes considérant que le PSO n'est pas reconnu pour offrir des performances satisfaisantes pour la résolution de grandes instances de problèmes (Tamayo-Vera et al., 2018; van den Bergh & Engelbrecht, 2004).

1.2. OBJECTIFS ET MÉTHODOLOGIE

L'objectif principal de cette thèse est de *Contribuer aux stratégies métaheuristiques à base de population pour résoudre efficacement des problèmes de grande taille*. Il est connu que la performance de plusieurs métaheuristiques se détériore rapidement avec une augmentation de la dimension du problème (Boussaïd et al., 2013). La structure comportementale du PSO et les possibilités d'adaptation de cette métaheuristiques en font une méthode de choix pour les expérimentations sur l'hybridation et la discrétisation (Hussain et al., 2019; Krause et al., 2013). Cet objectif principal est atteint par la proposition de méthodes hybrides et de procédés de discrétisation utilisant le PSO pour la résolution de grandes instances de problèmes. Trois objectifs secondaires servent à l'atteindre :

1. *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature ;*
2. *Adapter et comparer trois méthodes de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support ;*
3. *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA.*

Le premier objectif secondaire est atteint par la conception de trois modèles hybrides homogènes utilisant des variantes de PSO. Ces variantes sont le BPSO (James Kennedy, 2003), le CLPSO (Liang et al., 2006) et le CoLPSO (van den Bergh & Engelbrecht, 2004). Elles sont testées sur des problèmes continus classiques (Dixon & Szego, 1978; Molga & Smutnicki, 2005), de même que sur l'ensemble de références CEC'15 établi par Qu et al. (2016). Elles sont également comparées à un algorithme de la littérature nommé dynFWACM proposé par Yu et al. (2015).

Le deuxième objectif secondaire est atteint par l'étude et la comparaison de trois procédés de discrétisation sur une base commune. Ces procédés sont basés sur la priorité d'ordonnancement de H. Zhang et al. (2005), la liste de translations d'Anghinolfi et Paolucci (2009) et la liste de permutations de Clerc (2004). La base commune servant aux expérimentations consiste à appliquer ces trois procédés de discrétisation au PSO pour la résolution du SOP (Escudero, 1988) et du RCPSP (Kolisch & Sprecher, 1997) avec un ensemble de paramètres identiques.

Le troisième objectif secondaire est atteint par la combinaison des conclusions provenant des deux premiers objectifs secondaires pour la résolution d'un problème comportant des instances de grande taille, le MSA. Celui-ci est un problème de nature combinatoire NP-difficile (Lusheng Wang & Jiang, 1994) utilisé en bio-informatique pour effectuer, entre autres, la reconnaissance de gènes ou encore la reconstruction de l'arbre de vie (Pevsner, 2015). La complexité du MSA provient de la difficulté des algorithmes à produire un alignement le plus fidèle possible à la réalité et représentant adéquatement l'évolution des espèces. En combinant la meilleure hybridation et le procédé de discrétisation le plus performant, un nouveau comportement du PSO est proposé avec une codification de solutions sous forme d'anneau spécialisée en fonction du MSA, jumelé à un nouveau processus d'apprentissage par magnétisme. Les résultats de cet algorithme sont comparés

entre eux, de même qu'avec deux outils utilisés en bio-informatique : MAFFT (Katoh et al., 2002) et CLUSTAL (Hung et al., 2015; Larkin et al., 2007).

1.3. ORGANISATION DE LA THÈSE

Cette thèse est divisée comme suit. Les deux prochains chapitres regroupent la revue de la littérature et les trois suivants présentent les contributions effectuées dans cette thèse. Le Chapitre 2 débute avec une présentation de la littérature pour la compréhension des concepts et des enjeux reliés à la résolution de problèmes par les métaheuristiques. Après avoir établi ces éléments, ce chapitre définit deux stratégies de conceptions utilisées dans cette thèse, soient l'hybridation et la discrétisation.

Le Chapitre 3 poursuit avec la définition de la métaheuristique sélectionnée pour la résolution des problèmes : le PSO. Son fonctionnement et les derniers développements proposés dans la littérature y sont exposés. S'en suit une revue du PSO plus spécifique par la mise en évidence de l'hybridation et de la discrétisation utilisant cette métaheuristique. La problématique et les objectifs de la recherche sont par la suite décrits en détail afin de conclure la revue de la littérature.

Le Chapitre 4, vise à atteindre le premier objectif secondaire. Il débute avec la description des instances de problèmes continus permettant de valider les stratégies d'hybridation utilisant le PSO, soient les fonctions classiques et les fonctions provenant de l'ensemble CEC'15. La conception de trois hybridations homogènes utilisant plusieurs variantes de PSO sont ensuite présentées, comparées et analysées pour la résolution de ces problèmes.

Le Chapitre 5 poursuit avec la présentation de deux problèmes de support, soient le SOP et le RCPSP. Une description de trois procédés de discrétisation utilisés dans cette thèse et les expérimentations permettant de les comparer sur une base commune sont ensuite présentées. Le deuxième objectif secondaire est atteint en contribuant à la discrétisation du PSO par la validation du procédé de discrétisation permettant d'obtenir les performances moyennes les plus robustes.

L'atteinte du troisième objectif secondaire est réalisée par les propositions du Chapitre 6. Ce chapitre conclut les expérimentations avec la présentation des problèmes d'alignement de séquences. En utilisant les conclusions provenant des contributions du Chapitre 4 et du Chapitre 5, deux procédés de discrétisation sont présentés comportant une nouvelle modélisation de solutions en anneau. Un nouveau processus d'apprentissage par magnétisme est ensuite proposé pour l'algorithme hybride discret construit avec le PSO. L'algorithme ainsi formé est comparé et analysé pour la résolution du MSA et permet de valider la contribution au PSO pour la résolution de grandes instances de problèmes discrets.

Le dernier chapitre, le Chapitre 7, conclut cette thèse avec un retour sur les contributions et les expérimentations à la communauté scientifique. Il poursuit avec les perspectives de recherche permettant de pousser plus loin et de valider les propositions de cette thèse dans des contextes différents.

CHAPITRE 2 – LES STRATÉGIES DE CONCEPTION DE MÉTAHEURISTIQUES

2.1. INTRODUCTION

Ce chapitre aborde l'utilisation de stratégies de conception des algorithmes métaheuristiques pour la résolution de problèmes d'optimisation complexes et de grande taille. L'adaptation de ce type d'algorithmes pour résoudre des problèmes provenant de contextes réels ou théoriques est un sujet en constante évolution (Mahdavi et al., 2015).

À la suite de l'introduction des différents types de problèmes en optimisation et des algorithmes métaheuristiques, ce chapitre introduit une première stratégie pour la conception de métaheuristique : l'hybridation. Ce concept est défini avant de poursuivre sur les motivations dans la mise en œuvre de cette stratégie de conception. Trois classifications de méthodes hybrides sont décrites afin de mieux situer l'étendue des possibilités qu'offre une stratégie d'hybridation.

Ce chapitre introduit ensuite une deuxième stratégie offrant la possibilité d'adapter les métaheuristiques pour la résolution d'une plus grande variété de problèmes: la discrétisation. De nombreuses métaheuristiques, comme l'optimisation par essaims particulaires, ont été conçues à l'origine pour la résolution de problèmes continus. La discrétisation étend ainsi la portée de ces algorithmes pour résoudre des problèmes de nature différente, comme des problèmes de nature binaire ou discrète. La présentation des principes de base en discrétisation permet de mieux comprendre les techniques proposées dans la littérature portant sur ce sujet. Ces dernières œuvrent dans l'idée qu'une métaheuristique conçue pour résoudre des problèmes continus peut subir des modifications de comportement afin de pouvoir résoudre des problèmes discrets, et ce, sans perdre son essence (Krause et al., 2013).

Les algorithmes métaheuristiques représentent des stratégies de résolution devant être adaptées afin de constituer une approche pertinente et performante pour l'optimisation

d'un problème spécifique (El-Ghazali Talbi, 2009). Plusieurs choix doivent être faits lors de la conception d'une méthode de résolution en prenant en compte le type de problème visé. Les deux stratégies de conception énumérées, l'hybridation et la discrétisation, donnent un aperçu de l'étendue des options disponibles.

2.2. RÉSOLUTION DE PROBLÈMES PAR LES MÉTAHEURSITIQUES

2.2.1. PROBLÈMES D'OPTIMISATION

Une grande variété de problèmes d'optimisation se retrouve dans la littérature. Selon la théorie de la complexité, un problème d'optimisation est qualifié de NP-difficile lorsque le problème de décision associé est NP-complet (Garey & Johnson, 1979). Selon cet auteur, cette catégorie de problèmes est la plus difficile à résoudre parmi les problèmes dits « non déterministe polynomial » (*nondeterministic polynomial* - NP). De manière générale, un problème d'optimisation est également défini par trois éléments importants : un *espace de recherche* (variables de décision), une ou plusieurs *fonction(s) objectif* à optimiser et un ensemble de *contraintes* à respecter (Ravindran, 2016).

Le premier élément, l'*espace de recherche*, se compose de l'ensemble des différentes solutions au problème et une solution se représente par les valeurs sélectionnées pour les variables de décision. Ces variables peuvent être de diverses natures, telles que réelle, binaire, discrète et mixte, selon le problème à résoudre. Elles peuvent représenter par exemple une ville, un employé, la présence ou non d'un produit, ou encore les paramètres à fixer pour un système.

Le deuxième élément, la *fonction objectif*, se construit de manière à minimiser ou à maximiser une mesure de qualité selon les valeurs prises par les variables de décision. Elle

représente le but à atteindre pour le décideur. Un problème est dit *uni-objectif* s'il ne comporte qu'une seule fonction objectif pour le problème à résoudre. En contrepartie, un problème est dit *multi-objectif* s'il comporte deux fonctions objectif ou plus (Collette & Siarry, 2004).

Le troisième élément, les *contraintes*, peut contribuer à augmenter la complexité d'un problème à résoudre. L'ensemble des contraintes défini les conditions appliquées à l'espace de recherche et que les variables de décision doivent satisfaire. Boussaïd et al. (2013) mentionnent que les problèmes d'optimisation font face à des niveaux différents de contraintes : la nature des variables de décision, la constitution du problème et la modélisation du problème.

En premier lieu, les variables de décision ajoutent naturellement certaines contraintes à un problème étant donné leur nature continue ou discrète. La composition d'une solution pour un problème à variables réelles comporte un nombre infini de possibilités dans les limites des nombres réels pour chacune des variables de décision. Ces variables sont toutefois contraintes par l'espace de recherche sur lequel l'optimisation est exécutée. À l'opposé, les problèmes à variables entières comportent une contrainte naturelle par l'utilisation des nombres entiers. Selon cette définition des variables de décision, il est possible de distinguer plusieurs catégories de problèmes, dont les problèmes continus et les problèmes discrets. Ce travail s'intéresse à ces deux types de problèmes.

En deuxième lieu, il y a la constitution du problème étudié. Certains problèmes nécessitent l'utilisation d'un ensemble complet ou partiel de variables de décision afin de produire une solution valide. Par exemple, le problème du voyageur de commerce et celui d'ordonnancement séquentiel (*sequential ordering problem* - SOP) (Escudero, 1988) contiennent une contrainte de permutation, où toutes les variables de décision doivent être ordonnancées afin de constituer une solution. En contrepartie, le problème d'alignement

multiple de séquences (*multiple sequence alignment problem* – MSA) (Taylor, 1987; Lusheng Wang & Jiang, 1994) est un exemple de problème présentant une contrainte d'arrangement, où un sous-ensemble ordonné d'éléments provenant d'un ensemble plus grand peut constituer une solution.

En troisième lieu, certains problèmes apportent des contraintes additionnelles par leur modélisation. Le problème d'ordonnancement de projets avec contraintes de ressources (*resource-constrained projet scheduling problem* – RCPSP) (Pritsker et al., 1969) en est un exemple par l'ajout de contraintes informationnelles, matérielles, monétaires, humaines, ainsi que par l'utilisation de préséances entre les différentes tâches à ordonnancer. Certaines solutions peuvent donc être impossibles ou irréalistes et doivent être exclues des valeurs possibles de l'espace de recherche.

2.2.2. NATURE DES VARIABLES DE DÉCISION : LES PROBLÈMES CONTINUS ET DISCRETS

Boussaïd et al. (2013) définissent deux catégories particulières de problèmes selon la nature des variables de décision composant l'espace de recherche. Dans le cas où l'ensemble des variables est de nature continue, il sera alors question d'un problème continu. Dans un autre cas où l'ensemble des variables est de nature discrète, il sera alors question d'un problème discret. Cette section apporte davantage de précisions quant à la nature continue et discrète de ces problèmes.

Les problèmes continus utilisent une composition de solution dont les variables de décision proviennent du domaine des nombres réels. Certaines équations mathématiques font partie de cette catégorie, dont des fonctions bien connues telles que *Sphere*, *Ackley*, *Rosenbrock*, *Rastrigin*, ou encore *Griewank (Dixon & Szego, 1978; Molga & Smutnicki, 2005)*. Une série plus récente de problèmes continus proposent des fonctions

mathématiques avec un niveau de complexité plus élevé (Qu et al., 2016). Cette série, nommée *CEC'15*, accentue la complexité par l'ajout de rotations et de transformations de la fonction objectif. Certaines d'entre elles proposent un optimum unique. D'autres sont plutôt de nature multimodale, proposant alors plusieurs optimums locaux. L'optimisation par essaims particulaires (R. Eberhart & Kennedy, 1995), l'évolution différentielle (Storn & Price, 1997), l'algorithme des chauves-souris (Yang, 2010) et les stratégies d'évolution (Back, 1996) font partie des métaheuristiques conçues initialement pour la résolution de problèmes comportant des variables continues.

En ce qui concerne les problèmes discrets, la composition des solutions présente des variables de décision provenant du domaine des nombres entiers. Ces problèmes sont considérés de nature combinatoire étant donné l'explosion des possibilités de solutions lorsque le nombre de variables de décision augmente. Parmi ceux bien connus, Gass1 et Fu (2013) mentionnent le problème du voyageur de commerce. Celui-ci consiste à ordonnancer une liste de villes à parcourir les unes après les autres, tout en cherchant à minimiser la somme des distances parcourues. Parmi les métaheuristiques conçues pour la résolution de ce type de problème, on retrouve par exemple l'optimisation par colonies de fourmis (Dorigo & Gambardella, 1997), la recherche avec tabous (Glover, 1989, 1990), l'algorithme génétique (Goldberg & Holland, 1988), le système immunitaire artificiel (Farmer et al., 1986) et le recuit simulé (Kirkpatrick et al., 1983).

La taille des instances pour ces problèmes est un facteur difficile à gérer en optimisation. Il est significativement plus complexe de trouver un optimum global à une grande instance de problème, comparativement à un problème similaire utilisant une dimension plus petite. Une augmentation de la dimension implique une augmentation exponentielle de l'espace de recherche, et donc une quantité plus importante de combinaisons de solutions (van den Bergh & Engelbrecht, 2004). Quatre raisons contribuent

à la complexité non négligeable lors de l'utilisation d'un problème de grande taille (Mahdavi et al., 2015) :

- L'espace de recherche augmente exponentiellement avec l'ajout de variables de décision ;
- La faculté de résolution d'un problème d'optimisation se détériore avec l'augmentation de sa taille ;
- L'évaluation par la fonction objectif d'un problème de grande taille par une métaheuristique demande un coût de calculs considérable ; et
- Il est difficile d'évaluer le niveau d'interaction entre les variables de décision d'un problème.

La résolution des problèmes NP-difficiles favorise l'utilisation d'une méthode *approchée* afin de trouver une solution de qualité dans un temps raisonnable. Les métaheuristiques s'adressent particulièrement à la résolution de ces problèmes (Weise et al., 2009).

2.2.3. MÉTAHEURISTIQUES

Parmi les méthodes de résolution pour les problèmes continus et discrets, il y a une distinction entre les méthodes *exactes* et les méthodes *approchées*. Les méthodes *exactes* permettent de trouver la solution optimale lorsqu'une telle méthode est applicable. Le temps de résolution peut cependant devenir important avec une augmentation de la taille des instances à résoudre. Les méthodes *approchées* cherchent quant à elles à trouver des solutions de la meilleure qualité possible avec un temps de calcul moins sensible à la taille des instances à résoudre. Elles ne peuvent toutefois pas assurer l'optimalité des solutions trouvées. En ce sens, les méthodes *approchées* trouvent leur raison d'être dans le traitement des problèmes complexes et de grande taille. Ce type d'algorithme offre une faculté

d'adaptation permettant de résoudre efficacement une grande variété de problèmes (Whitacre, 2011).

Les *métaheuristiques* représentent une classe particulière d'algorithmes *approchés*. Elles sont définies par Osman et Kelly (1996) comme étant un processus itératif qui tire profit d'une heuristique subordonnée qui combine intelligemment différents concepts pour exploiter et explorer l'espace de recherche. Elles utilisent des stratégies d'apprentissage afin de structurer l'information. Tout ceci dans le but de trouver efficacement des solutions près de l'optimum, sans pouvoir le garantir.

La combinaison et l'ajustement entre l'*exploitation* et l'*exploration* de l'espace de recherche permettent d'obtenir une métaheuristique performante (El-Ghazali Talbi, 2009). L*'exploitation* peut être considérée comme un moyen d'optimiser localement une solution dans une région donnée. Une exploitation efficace consiste à extraire la meilleure solution parmi un ensemble de solutions voisines. À l'opposé, l'exploration constitue un moyen d'optimisation globale, où différentes régions de l'espace de recherche sont visitées. Le défi est d'équilibrer efficacement ces deux mécanismes lors de l'adaptation d'une métaheuristique pour la résolution d'un problème particulier.

Les métaheuristiques connaissent une popularité depuis quelques décennies (Hussain et al., 2019; Whitacre, 2011). Une grande variété de nouvelles métaheuristiques ont été proposées dans la littérature entre les années 2005 et 2015. Elles regroupent d'ailleurs les caractéristiques suivantes (A. Colorni et al., 1996) :

- Elles sont inspirées d'évènements ou de phénomènes provenant de la nature ;
- Elles ne sont pas déterministes ;
- Elles utilisent souvent une forme d'interaction entre différents agents, représentant implicitement une structure parallèle ; et
- Elles sont adaptatives à plus d'un problème.

El-Ghazali Talbi (2009) et Boussaïd et al. (2013) regroupent la variété de métaheuristiques selon deux grandes catégories : les métaheuristiques *à solution unique* et les métaheuristiques *à base de population*. La Figure 2-1 illustre la position des métaheuristiques parmi les différentes catégories de méthodes d'optimisation.

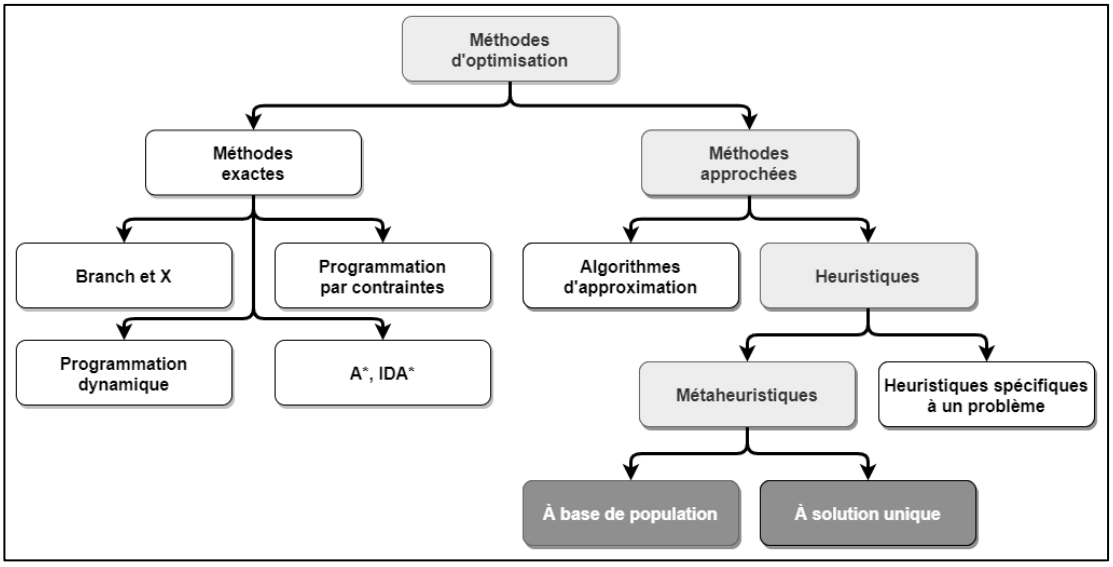

Figure 2-1 : Catégorisation des méthodes d'optimisation (adaptation de la figure de Talbi (2009)).

La première catégorie, *à solution unique*, consiste à travailler à chaque itération avec une seule solution pour la résolution d'un problème. Cette même solution évolue ensuite sur plusieurs itérations de l'algorithme afin d'obtenir une solution de bonne qualité. Cette catégorie contient plusieurs métaheuristiques bien connues axées davantage sur l'*exploitation* de l'espace de recherche (El-Ghazali Talbi, 2009). Par exemple, il y a le recuit simulé (Kirkpatrick et al., 1983), lequel est inspiré du refroidissement et du réchauffement de matériaux en métallurgie. Un autre exemple est la recherche avec tabous (Glover, 1989). Celui-ci utilise une forme de mémoire, appelée tabous, dans son processus de recherche de solutions. Les tabous permettent de restreindre l'accès à certaines solutions déjà rencontrées afin de forcer l'exploration de nouvelles solutions.

La deuxième catégorie, *à base de population*, regroupe les métaheuristiques composées d'un ensemble de solutions interagissant entre elles à chaque itération. Son fonctionnement suit trois étapes importantes axées davantage sur l'*exploration* de l'espace de recherche. Tout d'abord, une population de solutions est initialisée. Ensuite, une nouvelle population est générée via un processus propre à la métaheuristique. Finalement, cette nouvelle population est intégrée à la population précédente via un processus de sélection ou de remplacement. Les métaheuristiques à base de population peuvent se diviser en deux sous-catégories : les *algorithmes évolutionnaires* et les *algorithmes d'intelligence en essaims* (El-Ghazali Talbi, 2009).

La première sous-catégorie, les *algorithmes évolutionnaires*, consiste à faire évoluer une population de solutions en créant de nouvelles solutions à chaque itération de l'algorithme. Des solutions de la population courante sont sélectionnées et sont nommées parents. Ces solutions parents sont combinées afin de créer des solutions enfants via le principe de reproduction. Ces enfants remplacent ensuite des solutions provenant de la population précédente, afin de former une nouvelle population pour l'itération suivante. Le but est de continuellement améliorer la population par diverses stratégies d'évolution. Un exemple de métaheuristique évolutionnaire est l'algorithme génétique (Goldberg & Holland, 1988). Ce dernier consiste à construire de nouvelles solutions selon le principe de reproduction entre deux individus : une partie de la solution provient d'un parent, et une autre partie provient de l'autre parent.

La deuxième sous-catégorie, les *algorithmes d'intelligence en essaims*, regroupe les métaheuristiques reproduisant un comportement basé sur l'interaction entre les individus d'une espèce. Une caractéristique principale de cette sous-catégorie est que les individus coopèrent par un moyen de communication, et qu'ils effectuent des mouvements dans l'espace de recherche à la suite de cette communication. Un exemple de métaheuristique bien connue est l'optimisation par colonies de fourmis (Alberto Colorni et al., 1991; Dorigo &

Gambardella, 1997). Celle-ci reproduit le comportement de fourmis en quête de nourriture. Elle permet la communication entre les individus par le dépôt de phéromones sur le sol afin d'orienter la colonie. Un autre exemple de métaheuristique est l'optimisation par essaims particulaires (PSO) (R. Eberhart & Kennedy, 1995). Celle-ci reproduit le comportement d'un groupe d'oiseaux ou de bancs de poissons. Les individus communiquent au reste de la population la qualité des solutions rencontrées afin d'aiguiller les autres individus vers des optimums potentiels. Plus de détails sur cette métaheuristique sont disponibles au Chapitre 3.

Une grande partie des nouvelles métaheuristiques ayant été élaborées plus récemment sont recensées par Fister Jr et al. (2013), de même que par Xing et Gao (2014) et Hussain et al. (2019). Dans ces articles, il est noté que la quantité grandissante de métaheuristiques ouvre les possibilités sur diverses manières d'aborder un problème d'optimisation, spécifiquement depuis 2005. Hussain et al. (2019) mentionnent cependant que la plupart de ces nouvelles méthodes d'optimisation ne présentent que peu de contributions importantes au domaine des métaheuristiques. Considérant le théorème du « No Free Lunch » (Ho & Pepyne, 2002; Wolpert & Macready, 1997), aucune stratégie d'optimisation ne peut être assez performante pour être globalement la meilleure pour tous les problèmes existants. Un moyen efficace afin de se rapprocher de cet idéal est de combiner les forces de plusieurs métaheuristiques pour tirer profit de chacune d'entre elles. Cette combinaison d'éléments correspond à la première stratégie de conception, l'hybridation.

2.3. PREMIÈRE STRATÉGIE DE CONCEPTION : L'HYBRIDATION

2.3.1. QU'EST-CE QU'UNE MÉTHODE HYBRIDE?

En recherche opérationnelle, l'hybridation consiste à combiner en un seul processus deux méthodes d'optimisation ou plus. Certains de ces hybrides sont développés en combinant une méthode de résolution exacte avec une méthode de résolution approchée (Puchinger & Raidl, 2005). D'autres combinent plutôt plusieurs métaheuristiques entre elles. Il est possible de retrouver des exemples de ces hybridations dans la littérature, utilisant entre autres l'optimisation par colonie de fourmis, l'algorithme génétique, l'optimisation à voisinage variable, la recherche avec tabous, etc. (Raidl, 2015). Certains auteurs ont plutôt sélectionné le PSO afin de produire une nouvelle méthode hybride (Banks et al., 2008; Maitra & Chatterjee, 2008; Sha & Hsu, 2006). Plus de détails sur l'utilisation du PSO en hybridation sont donnés au Chapitre 3.

2.3.2. CLASSIFICATION DES MÉTAHEURISTIQUES HYBRIDES

Plusieurs types de modifications sont possibles pour assembler deux ou plusieurs algorithmes en un seul processus. Ces modifications peuvent être une communication de solution, ou encore un échange complet entre des segments d'un des algorithmes utilisés. Il est possible que la nature d'une des métaheuristiques soit conservée, tout comme il est possible qu'une des métaheuristiques utilisées subisse un changement au fonctionnement d'un de ses processus qui lui est propre et qui la définit. Certains ouvrages de la littérature établissent une taxonomie permettant de différencier les méthodes hybrides les unes des autres en aidant à les regrouper selon leurs ressemblances. En ce sens, trois classifications sont proposées. Une première selon E-G Talbi (2002), une deuxième selon Jourdan et al. (2009), et une troisième selon Raidl (2015).

E-G Talbi (2002) propose une recherche approfondie sur ce qui différencie les méthodes hybrides. Selon la taxonomie qu'il propose, la classification des différentes hybridations s'effectue principalement selon deux caractéristiques hiérarchiques : le niveau (*bas niveau/haut niveau*) et le comportement (*à relais/collaboratif*). Une hybridation de *haut niveau* signifie que chacune des métaheuristiques composant l'hybride ne contient pas de relation directe entre elles au sein de leur fonctionnement individuel. À l'opposé, une hybridation de *bas niveau* implique qu'une partie du fonctionnement d'une des métaheuristiques est partagée avec une partie de l'autre métaheuristique hybridée. Un comportement *à relais* implique quant à lui que le résultat provenant de l'exécution d'une des métaheuristiques est réutilisé afin d'initialiser les solutions pour l'exécution de la deuxième métaheuristique de l'hybride, tel un pipeline. À l'opposé, un comportement *collaboratif* implique que chaque métaheuristique est exécutée en parallèle, assurant une communication constante entre les méthodes afin d'obtenir une solution. Le résultat de ces caractéristiques hiérarchiques définit une taxonomie établie selon les quatre notations suivantes, représentées selon une composition de fonctions :

- $HRB\ (\mathrm{MH}_1\ (\mathrm{MH}_2)\ )$: (*Hybride à Relais de Bas niveau*)

  La métaheuristique MH_2 est intégrée en tant que composante de MH_1, où MH_1 est une métaheuristique à solution unique ;
- $HRH\ (\mathrm{MH}_1\ +\ \mathrm{MH}_2)$: (*Hybride à Relais de Haut niveau*)

  Les métaheuristiques MH_1 et MH_2 sont exécutées indépendamment l'une après l'autre ;
- $HCB\ (\mathrm{MH}_1\ (\mathrm{MH}_2)\ )$: (*Hybride Collaboratif de Bas niveau*)

  La métaheuristique MH_2 est intégrée en tant que composante de MH_1 ; et
- $HCH\ (\mathrm{MH}_1, \mathrm{MH}_2)$: (*Hybride Collaboratif de Haut niveau*)

  Les métaheuristiques MH_1 et MH_2 sont exécutées en parallèle et de manière indépendante.

Selon E-G Talbi (2002), il est possible de catégoriser toute hybridation entre métaheuristiques suivant cette taxonomie, en combinant une ou plusieurs d'entre elles. Afin d'illustrer sa proposition, l'auteur démontre l'application sur certains algorithmes de la littérature avec une représentation arborescente. Il mentionne entre autres un algorithme proposé par Boese, Kahng et Muddu (1994) dont le modèle est de type $HRH(LS + HCB(GA(LS))$. Pour cet exemple, l'arborescence de racine *HRH* indique qu'un algorithme de recherche locale (LS) est exécuté de manière indépendante, avant de poursuivre l'exécution ($+$) avec un algorithme génétique (GA) en utilisant les solutions obtenues par le LS pour l'initialisation de la population du GA. L'arborescence enracinée *HCB* indique, pour sa part, que le GA intègre au sein de son fonctionnement un LS afin d'obtenir une solution.

Des descriptifs sont également proposés par E-G Talbi (2002) afin de différencier les méthodes hybrides sous une même caractéristique hiérarchique. Alors que la caractéristique hiérarchique défini les éléments d'un hybride ayant rapport à la nature du problème traité, le descriptif défini plutôt comment l'algorithme est implémenté. Ces derniers incluent la nature *homogène/hétérogène*, l'optimisation *globale/partielle* et la fonction *spécialiste/générale* de l'hybride et font partie de la classification dite *à plat* de E-G Talbi. Ils permettent d'apporter une spécification lors de la classification des méthodes selon cette taxonomie. E-G Talbi utilise dans sa classification *à plat* (*flat*) le descriptif de la *nature* de ce qui est hybridé à l'aide des adjectifs *homogène* et *hétérogène*. Le terme *homogène* indique que les algorithmes ayant servis à l'hybridation proviennent de la même métaheuristique ou d'une variante de celle-ci. Le terme *hétérogène* indique quant à lui que les algorithmes hybridés proviennent de métaheuristiques différentes.

La taxonomie de E-G Talbi (2002) ne prend en considération que l'hybridation entre deux métaheuristiques. Jourdan et al. (2009) se sont basés sur les travaux de E-G Talbi afin de permettre également l'hybridation entre une métaheuristique et une méthode exacte. Ils étendent donc la taxonomie de E-G Talbi à une plus grande variété de méthodes de

résolution. Une partie de la classification selon les caractéristiques hiérarchiques de E-G Talbi est réutilisée, soient le niveau (*bas niveau/haut niveau*) et le comportement (*à relais/collaboratif*). Les quatre notations (*HRB*, *HRH*, *HCB* et *HCH*) sont également réutilisées. La différence entre la classification selon Jourdan et al., en comparaison avec celle selon E-G Talbi, provient de la partie à plat de la classification. Là où E-G Talbi définit trois descriptifs complémentaires comme étant la nature (*homogène*/*hétérogène*), l'optimisation (*globale*/*partielle*) et la fonction (*spécialiste*/*générale*), Jourdan et al. définissent plutôt les trois descriptifs suivants : le type de résolution (*exacte*/*approximée*), l'optimisation (*globale*/*partielle*) et la fonction de la coopération (*spécialiste*/*générale*).

Raidl (2015) propose une taxonomie alternative permettant également d'élargir l'applicabilité du travail original effectué par E-G Talbi (2002). Cette taxonomie, illustrée à la Figure 2-2, prend en considération quatre éléments clés : la *nature* de ce qui est hybridé, le *niveau* d'hybridation, l'*ordre d'exécution* et la *stratégie de contrôle* des éléments impliqués. Elle considère également l'existence des architectures parallèles dans la conception d'algorithmes, lesquelles s'avèrent de plus en plus utilisées en optimisation (Raidl, 2015).

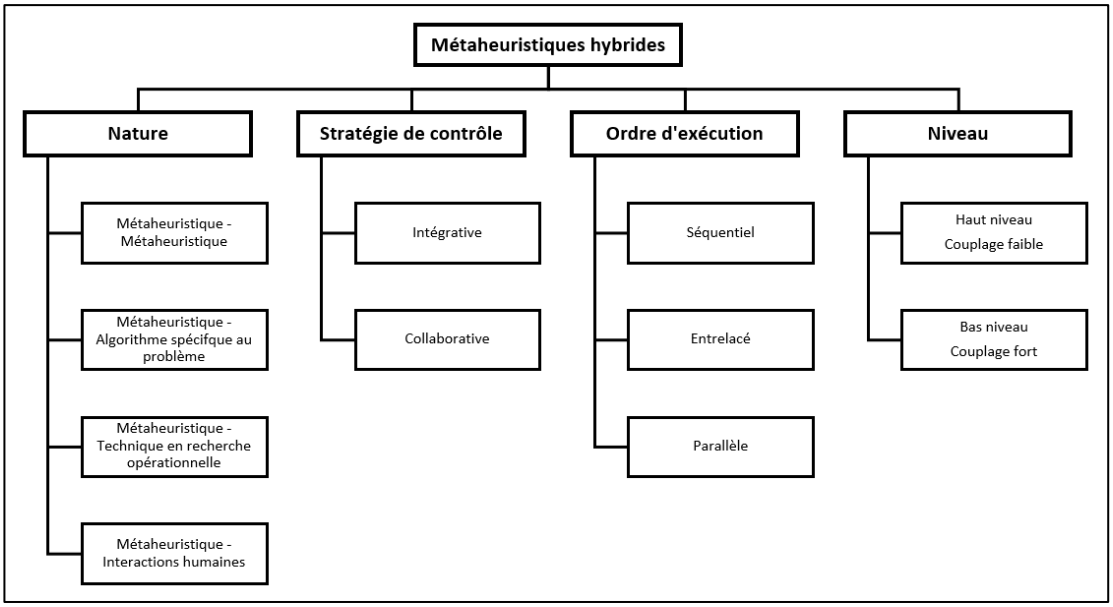

Figure 2-2 : Sommaire de la classification des métaheuristiques hybrides selon Raidl (2015)

2.4. DEUXIÈME STRATÉGIE DE CONCEPTION : LA DISCRÉTISATION

2.4.1. QU'EST-CE QUE LA DISCRÉTISATION?

Tel que mentionné précédemment, les métaheuristiques sont des méthodes génériques permettant de s'adapter à plusieurs problèmes d'optimisation. Elles se définissent tout de même à la base pour la résolution d'un type de problème précis. Par exemple, l'algorithme d'optimisation par essaims particulaires est conçu pour la résolution de problèmes continus. À l'opposé, l'algorithme d'optimisation par colonies de fournis est conçu pour la résolution de problèmes discrets. Il semblerait que la littérature recense un plus grand nombre de métaheuristiques traitant à la base des variables réelles que de métaheuristiques traitant des variables entières (Fister Jr et al., 2013; Hussain et al., 2019). Plusieurs d'entre elles sont adaptées afin qu'elles puissent servir à l'optimisation de problèmes à variables

entières. Ces adaptations correspondent à la deuxième stratégie de conception, la discrétisation.

La discrétisation consiste à trouver un procédé permettant de transformer une solution codifiée en données réelles pour une solution équivalente ou approximativement équivalente codifiée en données entières (Dougherty et al., 1995). Le but est de permettre à une méthode d'optimisation continue de résoudre un problème de nature discrète, tout en conservant une convergence de l'algorithme vers une solution de bonne qualité.

Les sections suivantes abordent tout d'abord la codification de solutions lors de la discrétisation de métaheuristiques. Par la suite, six différents procédés de discrétisation applicables aux métaheuristiques continues à base de population sont définis selon les travaux de Krause et al. (2013).

2.4.2. CODIFICATION DE SOLUTIONS

La discrétisation d'une métaheuristique est fortement tributaire de la définition des solutions pour le problème à l'étude. Cette codification de solutions détermine l'importance des modifications nécessaires lors de la discrétisation d'une méthode. Trois formes de codification de solutions sont décrites par Krause et al. (2013) : binaire, entière et réelles.

Les deux premières formes codifient la solution par l'utilisation de variables binaires ou entières. Dans ces deux cas, une métaheuristique conçue à la base pour la manipulation de vecteurs solutions composés de variables réelles, telle que le PSO, doit s'adapter au nouveau format de données dès l'initialisation des solutions. Les traitements suivant cette initialisation doivent également prendre en considération la nature des données à traiter. Les quatre opérations arithmétiques (+, -, x, /) utilisées habituellement lors de la manipulation

d'un vecteur solution doivent alors être redéfinies puisque ces opérations ne peuvent avoir le même comportement selon la nature des données.

La troisième forme permet de conserver le fonctionnement d'une métaheuristique conçue initialement pour le traitement de solutions en variables réelles. Dans ce cas, la codification d'une solution demeure en variables réelles malgré la discordance avec le problème à résoudre. Une discrétisation utilisant cette forme utilise plutôt une méthode de transformation de solutions afin d'effectuer l'évaluation de la solution par la fonction objectif. Cette transformation s'effectue selon la nature de la solution du problème, comme par exemple, du réel au discret et du réel au binaire. Un des avantages de cette forme de codification est qu'elle permet de conserver l'intégralité de la méthode et de l'essence du fonctionnement de la métaheuristique. En contrepartie, la solution doit être transformée afin d'avoir une signification dans le cadre du problème à résoudre.

2.4.3. DISCRÉTISATION DE MÉTAHEURISTIQUES À BASE DE POPULATION

Krause et al. (2013) recensent six différents procédés de discrétisation applicables aux métaheuristiques continues à base d'intelligence en essaims. Les trois formes de codification de solutions énumérées précédemment sont utilisées pour les définir. Les deux premiers procédés servent à transformer les variables de décision en solution binaire. Les trois procédés subséquents servent à transformer les variables de décision en solution entière. Le sixième et dernier procédé regroupe les moyens conçus spécifiquement pour l'utilisation de l'optimisation par essaims particulaires.

Le premier procédé provient d'une étude de J. Kennedy et Eberhart (1997). Il consiste à utiliser une fonction sigmoïdale permettant la transformation des variables réelles contenues à chaque dimension d d'un vecteur solution en variables binaires. Cette transformation est appliquée au vecteur solution q_i^t d'un individu i de la population lors de

l'évaluation par la fonction objectif à l'itération t. Afin de former un vecteur solution binaire, chacune des dimensions de q_i^t est transformée en 0 ou en 1. L'Équation 2-1 est un exemple de fonction utilisée $(alea)$ afin de générer un nombre réel aléatoire entre 0 et 1. Si ce nombre est inférieur au résultat de la fonction sigmoïdale présente à l'Équation 2-1, le chiffre 1 est sélectionné. Autrement, le chiffre 0 est sélectionné. Par exemple, posons un vecteur solution $q_i^t = (0.90, 0.35,0.03,0.21,0.17)$ et un nombre aléatoire 0.37 pour la première dimension. Considérant que la fonction sigmoïdale pour 0.90 résulte en un nombre près de 0.71, et que 0.37 est plus petit que 0.71, alors la valeur 1 est inscrite à la première dimension du vecteur solution.

$$q_{id}^t = \begin{cases} 1 & \text{si } alea() \leq \frac{1}{1+exp(-q_{id}^t)} \\ 0 & \text{sinon} \end{cases} \qquad (2\text{-}1)$$

Le deuxième procédé de discrétisation est également utilisé pour la transformation de variables réelles en variables binaires. L'équation servant à la transformation des solutions est cependant différente de celle provenant du premier procédé. La dimension d au sein du vecteur solution q_i^t comportant la plus haute valeur est enregistrée à la première dimension du vecteur résultant $q_i^{t\prime}$. L'index de la deuxième plus haute valeur au sein de q_i^t est enregistrée à la suite, en répétant le processus jusqu'à ce que toutes les dimensions du vecteur solution aient été traitées. Chacune des variables de $q_i^{t\prime}$ est ensuite transformée en binaire à l'aide de l'Équation 2-2, en vue de l'évaluation par la fonction objectif. Il y a transformation d'une variable du vecteur solution en chiffre 1 lorsque cette valeur est strictement plus haute que la valeur inscrite à la dimension subséquente. La valeur 0 est inscrite dans le cas contraire. Par exemple, un vecteur solution $q_i^t = (0.90, 0.35,0.03,0.21,0.17)$ permet de générer le vecteur résultant $q_i^{t\prime} = (1, 2, 4, 5, 3)$. Celui-ci devient, après l'utilisation de l'Équation 2-2, le vecteur $q_i^{t\prime\prime} = (0, 0, 0, 1, 0)$.

$$q_{id}^{t\,\prime\prime} = \begin{cases} 1 & \text{si } q_{id}^{t\,\prime} > q_{i\,d+1}^{t\,\prime} \\ 0 & \text{sinon} \end{cases} \quad (2\text{-}2)$$

Le troisième procédé consiste à transformer une solution codifiée en variables réelles vers une permutation de variables discrètes. Chacune des valeurs du vecteur solution q_i^t est délimitée dans l'ensemble des réels entre 0 et 1. Un vecteur résultant $q_i^{t\,\prime}$ est formé en vue de l'évaluation par la fonction objectif, où chacune des dimensions d de ce dernier vecteur est composée d'un nombre entier déterminé selon le tri croissant des variables contenues dans q_i^t. La dimension de q_i^t présentant la plus basse valeur parmi l'ensemble du vecteur solution se voit attribuer le chiffre 1 à la dimension correspondante dans le vecteur résultant $q_i^{t\,\prime}$. La deuxième valeur la plus basse est ensuite associée au chiffre 2. Ce processus est répété jusqu'à ce que toutes les valeurs contenues dans le vecteur solution soient associées à des nombres entiers, en incrémentant de 1 la valeur assignée à chaque fois. Par exemple, un vecteur solution $q_i^t = (0.90, 0.35, 0.03, 0.21, 0.17)$ devient après transformation le vecteur résultant $q_i^{t\,\prime} = (5, 4, 1, 3, 2)$.

Le quatrième procédé de discrétisation est similaire au précédent. Il consiste à trouver la plus petite variable contenue dans le vecteur solution q_i^t et d'inscrire l'index (la dimension) de cette variable comme premier élément du vecteur résultant $q_i^{t\,\prime}$. La dimension où est située la deuxième variable la plus basse est ensuite inscrite en deuxième position du vecteur résultant. Ce processus est répété jusqu'à ce que toutes les variables du vecteur solution soient parcourues. En réutilisant l'exemple du vecteur solution q_i^t du procédé de discrétisation précédent, le vecteur solution devient après transformation le vecteur résultant $q_i^{t\,\prime} = (3, 5, 4, 2, 1)$.

Le cinquième procédé de discrétisation consiste à arrondir les variables réelles contenues à l'intérieur du vecteur solution q_i^t afin de les transformer en variables entières. Pour construire le vecteur résultant $q_i^{t\,\prime}$, les décimales incluses dans chaque dimension d du

vecteur solution sont tronquées ou arrondies à l'entier le plus près. Le résultat de cette opération compose le vecteur résultant $q_i^t{}'$. Par exemple, un vecteur solution $q_i^t = (1.22, 5.34, 18.12, 9.60, 0.41)$ devant subir une troncature de ses décimales devient, après transformation, le vecteur résultant $q_i^t{}' = (1, 5, 18, 9, 0)$.

Le sixième et dernier procédé de discrétisation est utilisé dans le cas où la métaheuristique subissant une adaptation est le PSO. Il consiste à redéfinir le processus de déplacement d'une particule, considérant que cette métaheuristique propose la manipulation de vecteurs afin de mettre à jour les solutions. Selon Krause et al. (2013), près de 25% des algorithmes continus à base de population sélectionnés pour la résolution de problèmes combinatoires utilisent le PSO. La Section 3.4 aborde avec plus de détails les procédés de discrétisation du PSO.

2.5. CONCLUSION

Ce chapitre fait état de l'avancement des métaheuristiques et de leurs stratégies de conception lors de la résolution de problèmes. Un accent plus particulier a été mis sur la différenciation entre la nature continue et discrète des problèmes d'optimisation, de même que sur les différentes catégories de métaheuristiques retrouvées dans la littérature. Il y a eu une distinction entre les méthodes *exactes* et les méthodes *approchées*. Les métaheuristiques font partie des méthodes approchées et se subdivisent en deux catégories : les algorithmes *à solution unique* et les algorithmes *à base de population*. Cette deuxième catégorie, à base de population, se subdivise également en deux sous-catégories, soient les algorithmes d'intelligence en essaims et les algorithmes évolutionnaires. Les différentes contributions de la littérature à ces méthodes d'optimisation a laissé place à des stratégies de conception permettant d'adapter une plus grande variété d'algorithmes pour la résolution de problèmes, et ce, peu importe la nature continue ou discrète des variables de décision.

La première stratégie de conception abordée, l'hybridation, a été décrite. Plusieurs classifications de métaheuristiques hybrides, soient celle de E-G Talbi (2002), de Jourdan et al. (2009) ainsi que celle de Raidl (2015), sont énumérées afin d'évaluer l'étendue des possibilités qu'offre l'hybridation. La deuxième stratégie de conception, la discrétisation, est par la suite décrite. L'emphase a été mis sur l'impact de la codification d'une solution, de même que sur les six procédés de discrétisation élaborés par Krause et al. (2013) pour l'adaptation d'une métaheuristique continue à *base de population*.

Les procédés énoncés permettent de guider l'adaptation de méthodes d'optimisation telle que le PSO. Cet algorithme s'avère être la métaheuristique la plus utilisée dans un contexte de discrétisation (Krause et al., 2013). La combinaison des deux stratégies de conception permet au PSO d'améliorer sa capacité à résoudre une plus grande variété de problèmes. Cette métaheuristique constitue le sujet principal du Chapitre 3.

CHAPITRE 3 - L'OPTIMISATION PAR ESSAIMS PARTICULAIRES (PSO) ET SES VARIANTES

3.1. INTRODUCTION

L'optimisation par essaims particulaires (R. Eberhart & Kennedy, 1995; Y. Shi & Eberhart, 1998) est une métaheuristique d'intelligence en essaim, appartenant à la branche des algorithmes à base de population (El-Ghazali Talbi, 2009). Un essaim, composé d'une population de particules, utilise la coopération afin que chacun des individus contribue à l'amélioration de la communauté. Le comportement de l'algorithme est basé sur les interactions d'un groupe d'oiseaux (particules) en quête de nourriture. Par analogie, l'essaim se déplace au-dessus d'un espace de recherche en évaluant la position de chacune des particules selon la fonction objectif du problème étudié. Il y a amélioration des solutions par l'interaction apportée entre les particules et leurs déplacements à travers les itérations de l'algorithme. Le PSO est conçu pour la résolution de problèmes continus et permet d'obtenir d'excellentes performances pour la résolution de ce type de problèmes (Banks et al., 2008).

Le présent chapitre décrit le PSO en présentant le comportement des particules pour la résolution de problèmes, de même que l'importance des paramètres pour la calibration de cette métaheuristique. Par la suite, différentes variantes du PSO sont décrites, avant de poursuivre avec des hybridations de la littérature ayant été développées avec le PSO. Le présent chapitre poursuit avec les différents procédés de discrétisations appliqués au PSO pour la résolution de problèmes discrets. Une attention particulière est portée sur la manière qu'un procédé doit être défini pour effectuer une discrétisation fonctionnelle du PSO. Le chapitre conclut avec les objectifs de la recherche pour cette thèse, ainsi que les moyens utilisés afin de démontrer les contributions dans les stratégies métaheuristiques par l'utilisation du PSO lors de la résolution de grandes instances de problèmes.

3.2. FONCTIONNEMENT DU PSO

3.2.1. PRINCIPES DE BASE

Anghinolfi et Paolucci (2009) définissent les composantes d'une particule comme suit : chaque particule i d'un essaim de grandeur N est composée d'une position x_i, d'une mémoire p_i et d'une vélocité v_i. La position x_i représente une solution au problème posé grâce aux coordonnées où est localisée la particule au sein de l'espace de recherche. Cet espace de recherche a une dimension de taille D. Ainsi, $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, où x_{iD} est la $d - ième$ composante, et $d = 1,2,..,D$. Autrement dit, chacune des dimensions représente une variable de décision du problème à résoudre. Une dimension de grandeur D engendre donc un problème à résoudre comportant D variables de décision. La mémoire p_i sert à retenir la meilleure position (solution) visitée par une particule à travers les itérations de l'algorithme, selon le résultat provenant de l'évaluation par la fonction objectif. Elle est notée par $p_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, et elle est enregistrée durant l'exploration de l'espace de recherche. La vélocité v_i sert à connaître la direction et la vitesse avec laquelle la particule se déplace au sein de l'espace de recherche. Elle est notée par $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ et elle est représentée par un vecteur de déplacement à appliquer à la position d'une particule pour changer son emplacement. Les notations x_i^t et v_i^t sont utilisées afin de représenter la position et la vélocité d'une particule i à l'itération t de l'algorithme.

En plus de ces informations, chaque particule peut communiquer sa mémoire p_i à la totalité ou à un sous-ensemble de la population. Cet échange « social » entre les particules s'effectue via un vecteur d'information générique représentant la meilleure position de la collectivité (de l'ensemble des particules). Cette composante est représentée par $g = (g_1, g_2, \ldots, g_D)$. Elle représente également la meilleure solution au problème et contiendra les variables de décision de la solution finale une fois la résolution complétée.

Les composantes x_i, p_i, v_i et g servent pour le calcul du déplacement des particules de l'essaim et pour leurs mises à jour lors de la résolution d'un problème. La combinaison de la position, de la vélocité, de la mémoire et de la meilleure solution collective définit le contenu des deux équations principales régissant le PSO : l'Équation 3-1 et l' Équation 3-2. L'Équation 3-1 met à jour la vélocité d'une particule avant que l'Équation 3-2 mette à jour sa nouvelle position selon la nouvelle vélocité calculée avec l'équation précédente. Deux nombres réels aléatoires r_1 et r_2 sont utilisés pour y parvenir, où $r_1, r_2 \in [0,1]$. Trois paramètres au PSO, w, c_1 et c_2, sont également nécessaires afin de pondérer les composantes énumérées précédemment. Ces paramètres sont détaillés à la Section 3.2.2.

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (p_i - x_i^t) + c_2 \cdot r_2 \cdot (g - x_i^t) \quad (3\text{-}1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (3\text{-}2)$$

L'application de ces deux équations est représentée par la Figure 3-1 suivante. Elle est une adaptation de l'image de Clerc (2006). Elle illustre le déplacement d'une particule de la position x_i^t à l'itération *t* vers la position x_i^{t+1} pour l'itération suivante, par la combinaison de l'influence des composantes p_i, g et v_i^t.

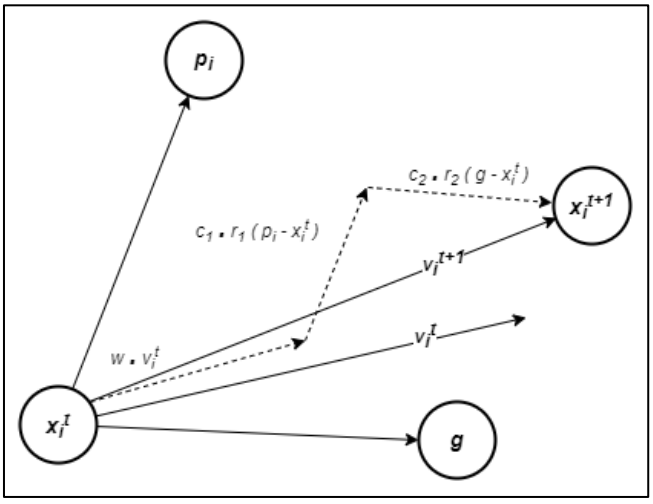

Figure 3-1 : Illustration du déplacement d'une particule du PSO lors de sa mise à jour (adaptation de Clerc (2006))
Figure libre accès

La Figure 3-2 illustre le fonctionnement du PSO. Après l'initialisation des composantes de base, il y a l'encadré 3-2A mettant en évidence la mise à jour des D dimensions d'une particule i. Une itération sur chaque dimension (variable de décision) d est effectuée afin de calculer les nouvelles valeurs contenues dans la nouvelle vélocité v_{id}^{t+1} et dans la nouvelle position x_{id}^{t+1}. Une vélocité v_{id}^{t+1} ne peut pas être inférieure à v_{min} ou supérieure à v_{max}. Cette limitation empêche une particule de bouger trop vite au sein de l'espace de recherche et ainsi de nuire à l'exploitation des solutions en ralentissant la convergence de l'algorithme. Dans le même ordre d'idée, une position x_{id}^{t+1} ne peut pas être inférieure à x_{min} ou supérieure à x_{max}. Ces constrictions sur la vélocité et sur la position d'une particule sont définies par les limites de l'espace de recherche. Une fois la mise à jour des D dimensions complétées, le contenu de la nouvelle position générée est évalué à l'aide de la fonction objectif $f(.)$ dans l'encadré 3-2B. Cet encadré regroupe les validations nécessaires vérifiant si la solution s'est améliorée pendant l'itération courante. Il y a donc vérification si la nouvelle position calculée x_i^{t+1} offre une qualité de solution supérieure selon la mémoire p_i et selon la meilleure position collective g. Le processus est par la suite répété pour l'ensemble des N particules de l'essaim et pour un nombre maximal d'itérations T.

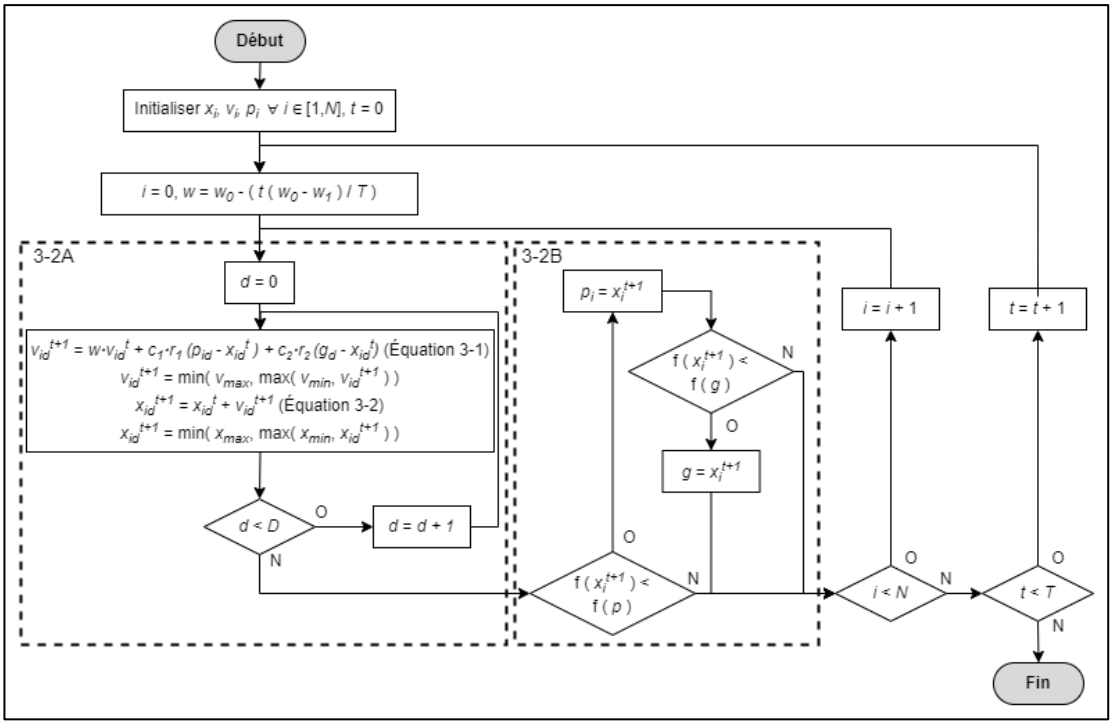

Figure 3-2 : Organigramme du fonctionnement du PSO. 3-2A Mise à jour d'une particule *i*. 3-2B Évaluation et mise à jour des meilleures solutions (cognitive et globale).
© Hugo Deschênes

Le modèle de PSO présenté dans ce chapitre ne comporte pas de sous-ensemble de particules et correspond à la variante nommée GPSO (Global PSO) (Clerc, 2006). Dans cette variante, chacun des individus communique sa position à l'ensemble de la communauté afin d'établir la meilleure solution globale. Cette meilleure position globale g agit en tant que centre d'oscillation des particules de l'essaim, lequel est le point focal autour duquel toute la population gravite afin de tenter une amélioration des solutions. L'ouvrage de Clerc (2006) présente également des moyens alternatifs permettant d'établir la communication entre les particules afin de transmettre les meilleures positions individuelles au reste de l'essaim. En effet, il est possible d'effectuer des regroupements où les particules n'échangent l'information qu'avec, par exemple, certaines particules voisines (informatrices) selon une topologie de communication (circulaire, aléatoire, etc.). Cette alternative est nommée LPSO (Local PSO). Par exemple, dans une topologie de communication circulaire, chaque particule se voit associer un successeur et un prédécesseur. Chaque particule n'a donc accès qu'à deux

informatrices : les deux particules voisines selon la distribution établie lors de l'initialisation des composantes. García-Nieto et Alba (2012) ont par la suite apporté une modification à la quantité d'informatrices à utiliser pour le LPSO, proposant que le nombre idéal soit plutôt de 6±2 lors de la résolution de problèmes à variables réelles. J. Kennedy et Mendes (2002) proposent plutôt que la quantité idéale soit de 4. La différence de performance apportée quant à l'utilisation du LPSO en comparaison avec le GPSO étant minime (Banks et al., 2007), le PSO est généralement préféré en format global dû principalement à sa simplicité d'implémentation, en omettant d'inscrire le « G » à son acronyme.

3.2.2. DÉFINITION DES PARAMÈTRES

La taille de la population N est un paramètre ayant besoin d'être fixé lors de l'initialisation du PSO. Le nombre sélectionné de particules peut influencer grandement l'efficacité de la résolution en modifiant la capacité d'*exploitation* et d'*exploration* de l'espace de recherche. Une étude mentionne qu'il est commun d'observer une taille de population variant entre 20 et 50 particules, où un plus grand nombre de particules est préférable pour la résolution de problèmes de grande taille (Poli et al., 2007).

Afin de bien représenter le facteur social et collectif des interactions, chacune des composantes d'une particule est pondérée par des paramètres permettant de varier le « niveau de confiance » envers elles. Le but de l'utilisation de ces paramètres est d'aider une particule à s'orienter en modifiant la confiance envers sa vélocité actuelle v_i, sa mémoire p_i et la meilleure solution collective g. Y. Shi et Eberhart (1998) ont ajouté un facteur de confiance en soi nommé w pondérant la vélocité v_i avec laquelle la particule se déplace. Ce facteur est également nommé « poids d'inertie » dans la littérature. Un facteur de confiance en sa mémoire c_1 est défini afin de pondérer la meilleure position p_i visitée par une particule au sein de l'espace de recherche. Un facteur de confiance envers les autres particules c_2 est également défini, lequel sert à pondérer la position rencontrée par la meilleure solution g de

l'essaim. L'utilisation de tous ces paramètres pondérant les vecteurs de déplacement influence le nouvel emplacement d'une particule, c'est-à-dire, sa prochaine position (solution). Clerc (1999) démontre que la rapidité de convergence des particules dépend fortement du choix de ces trois paramètres (w, c_1 et c_2).

Les paramètres du PSO ont évolué depuis sa conception. Dans la définition du PSO proposée par R. Eberhart et Kennedy (1995), seulement deux facteurs sont définis de façon à intégrer l'aléatoire à l'algorithme : c_1 et c_2. Afin d'assurer le bon fonctionnement de cette métaheuristique, les auteurs proposent la constriction des valeurs de la vélocité d'une particule selon les valeurs minimales et maximales de l'espace de recherche. R. C. Eberhart et Shi (2000) ont par la suite déterminé un calcul permettant d'introduire cette constriction sous la forme d'un nouveau facteur noté W (laquelle diffère du paramètre d'inertie w). L'Équation 3-3 calcule la valeur de cette constriction W, où $\varphi = c_1 + c_2$ et $\varphi > 4$. L'Équation 3-4 permet la mise à jour de la vélocité telle que proposé initialement par Russ C. Eberhart et Shi. Ces auteurs ont par la suite proposé une version améliorée et simplifiée, où le facteur W devient le paramètre w. L'Équation 3-4 est le calcul préliminaire de mise à jour de la vélocité d'une particule ayant servi à l'élaboration de l'Équation3-1, laquelle est utilisée maintenant dans la formulation classique du PSO.

$$W = \frac{2}{\left|2-\varphi-\sqrt{\varphi^2-4\varphi}\right|} \tag{3-3}$$

$$v_i^{t+1} = W \cdot \left(v_i^t + c_1 \cdot r_1 \cdot (p_i - x_i^t) + \ c_2 \cdot r_2 \cdot (g - x_i^t)\right) \tag{3-4}$$

R. C. Eberhart et Shi (2000) suggèrent de fixer les paramètres $c_1 = c_2 = 1.49618$, de même que le paramètre d'inertie $w = 0.7298$. Van den Bergh et Engelbrecht (2006) proposent une étude comportementale des particules afin d'analyser la rapidité de convergence de celles-ci et pour valider la fixation des paramètres. Leurs calculs mettent en évidence cinq combinaisons idéales pour les trois paramètres comportementaux du PSO, où

quatre de ces combinaisons proviennent de leurs recherches et une autre provient de R. C. Eberhart et Shi :

1- $w = 0.001, c_1 = c_2 = 2.0$

2- $w = 0.7298, c_1 = c_2 = 1.49618$ (R. C. Eberhart & Shi, 2000)

3- $w = 0.7, c_1 = c_2 = 1.4$

4- $w = 0.7, c_1 = c_2 = 2.0$

5- $w = 0.9, c_1 = c_2 = 2.0$

Selon leurs tests sur différentes fonctions mathématiques appartenant au domaine de l'optimisation continue, la deuxième combinaison de paramètres s'avère être la meilleure. Ces recherches démontrent la sensibilité des paramètres du PSO, de même que l'importance du choix des valeurs à assigner à w, c_1 et c_2.

Au lieu de fixer la valeur de l'inertie w en début d'exécution, une approche proposée dans la littérature consiste à adapter progressivement sa valeur selon le nombre d'itérations de l'algorithme. R. C. Eberhart et Shi (2001) suggèrent d'utiliser une équation permettant de réduire la valeur de l'inertie de manière linéaire sur l'ensemble des itérations de l'algorithme. Il est démontré qu'utiliser cette technique permet une amélioration de la qualité des solutions du PSO. Un poids d'inertie initial w_0 est fixé à une valeur de 0.9 et un poids d'inertie final w_1 est fixé à une valeur de 0.4 avec une variation linéaire. Le but de cette technique est de favoriser l'exploration de l'espace de recherche au début de l'algorithme et de favoriser l'intensification des solutions à la fin. Cette décroissance linéaire est l'approche la plus couramment utilisée dans la littérature (Jordehi & Jasni, 2015). Une autre approche consistant à appliquer une diminution du poids d'inertie selon l'Équation 3-5 est également proposée par Ratnaweera et al. (2004). Cette dernière dépend du nombre d'itérations t où l'essaim en entier est mis à jour et selon le nombre maximal d'itérations T spécifié comme paramètre à l'algorithme. La valeur de w n'est donc pas mise à jour de manière linéaire. Elle

s'adapte plutôt pour débuter avec une grande valeur pour favoriser l'exploration de l'espace de recherche, et termine avec une petite valeur pour favoriser l'exploitation.

$$w = w_0 - \left(\frac{t(w_0 - w_1)}{T}\right) \quad (3\text{-}5)$$

La nature et la quantité de paramètres peuvent varier si une variante de PSO est utilisée plutôt que la métaheuristique originale. Certaines variantes n'utilisent aucun paramètre alors que d'autres en ajoutent. La section suivante aborde les particularités reliées à l'utilisation de certaines variantes de PSO.

3.2.3. VARIANTES DU PSO

Trois variantes du PSO sont décrites dans les sous-sections suivantes. Ces variantes sont présentées pour le gain de performance qu'elles apportent dans l'amélioration du PSO selon leurs auteurs respectifs et par leur conception organiquement bien différente. Chacune d'elles apporte une différence au niveau de l'exploration de l'espace de recherche et de l'intensification des solutions en modifiant les équations de mise à jour d'une particule, en ajoutant des processus en cours de calculs ou en modifiant la structure de l'essaim de particules. Les trois variantes présentées sont le *Barebones* PSO (James Kennedy, 2003), le *Comprehensive Learning* PSO (Liang et al., 2006) et le *Cooperative Learning* PSO (van den Bergh & Engelbrecht, 2004).

3.2.3.1. *BAREBONES* PSO (BPSO)

Tel que vu précédemment, le PSO est une métaheuristique utilisant un certain nombre de paramètres. Ceci apporte un enjeu supplémentaire à l'optimisation en cherchant à ajuster adéquatement ces éléments pour obtenir les meilleurs résultats possibles. Le BPSO vient pallier cet enjeu en proposant une variante avec moins de paramètres à définir, sans

compromettre la qualité des résultats (James Kennedy, 2003). Son utilisation en est donc grandement simplifiée.

Pour fonctionner, le BPSO élimine le concept de vélocité afin de créer une génération statistique de positions dans l'espace de recherche. Dans la formulation du PSO, la meilleure position globale g agit en tant que centre d'oscillation pour les autres particules de l'essaim lors de la mise à jour de leur vélocité. Lorsque les particules se retrouvent trop près les unes des autres, leur vélocité diminue significativement. Elles finissent par avoir une vélocité presque nulle lorsqu'elles convergent. Bien qu'elles contribuent efficacement à l'exploitation de l'espace de recherche, elles nuisent également à améliorer l'exploration en fin de résolution.

La génération statistique utilisée pour les particules suit une distribution normale basée sur les solutions du problème, en calculant la moyenne et la variance des variables de décision. L'Équation 3-2 de mise à jour de la position d'une particule est remplacée par une équation générant un nombre aléatoire provenant d'une distribution normale. Cette mise à jour calcule une moyenne μ selon l'Équation 3-6 et un écart-type σ selon l'Équation 3-7 pour chacune des dimensions d d'une particule i. La position de la particule est ensuite mise à jour par l'utilisation de l'Équation 3-8(3-8 en générant un nombre aléatoire provenant d'une distribution normale par l'utilisation de la moyenne et de la variance.

$$\mu = 0.5 \cdot (p_{id} + g_d) \quad (3\text{-}6)$$

$$\sigma = |p_{id} - g_d| \quad (3\text{-}7)$$

$$x_{id}^{t+1} = normale(\mu, \sigma) \quad (3\text{-}8)$$

Un résumé du fonctionnement du BPSO est illustré à la Figure 3-3, où $f(.)$ représente la fonction objectif servant à évaluer une position et T le nombre maximal d'itérations. L'algorithme débute avec l'encadré 3-3A regroupant les éléments de la mise à

jour d'une particule i du BPSO, en calculant tout d'abord μ et σ. Il y a ensuite la génération de la nouvelle position x_{id}^{t+1} pour les D dimensions composant la particule. L'encadré 3-2B qui suit est identique à celui retrouvé au sein du PSO de la Figure 3-2. Il regroupe le processus permettant d'évaluer la particule avec la fonction objectif $f(.)$ lors d'une itération t, de même que la mise à jour de la meilleure position cognitive p_i et de la meilleure position globale g.

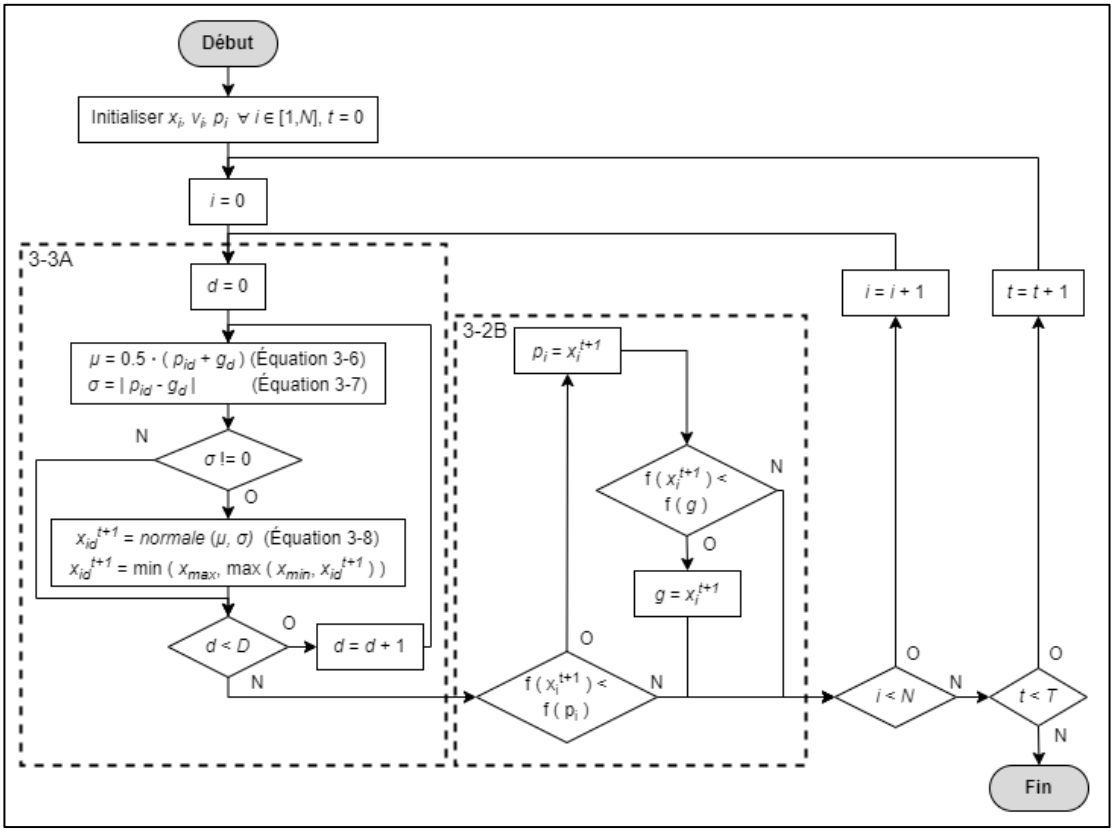

Figure 3-3 : Organigramme du fonctionnement du BPSO. 3-3A Mise à jour d'une particule *i*. 3-2B Évaluation et mise à jour des meilleures solutions (cognitive et globale).
© Hugo Deschênes

Des versions alternatives au BPSO existent dans la littérature. Alors que le BPSO utilise une distribution gaussienne afin de mettre à jour la position d'une particule, une autre méthode utilise plutôt une distribution de Lévy (Richer & Blackwell, 2006). Certains auteurs ajoutent une composante de saut lors de la mise à jour de la particule en permettant aux

solutions stagnantes de sauter à un autre endroit dans l'espace de recherche (al-Rifaie & Blackwell, 2012). Un processus similaire avec quelques différences mineures est également proposé où il y a plutôt utilisation d'une distribution de Cauchy pour modifier la composition d'une particule (Krohling & Mendel, 2009). Certains auteurs utilisent également le BPSO pour l'hybridation avec d'autres variantes de PSO. Par exemple, B. Jiang et Wang (2014) hybrident le BPSO avec l'algorithme de coévolution coopérative afin de résoudre le problème de partitionnement de groupes (*partitional clustering problem*).

3.2.3.2. *COMPREHENSIVE LEARNING* PSO (CLPSO)

Dans la version classique du PSO, chaque particule continue d'être attirée vers la meilleure position globale g, et ce, peu importe la qualité de la solution. Le CLPSO est une variante de PSO qui aide à pallier ce problème. Grâce au déclenchement d'un processus d'apprentissage compréhensif, il permet aux particules de continuer d'explorer efficacement l'espace de recherche lorsque les solutions stagnent pendant trop de générations (Liang et al., 2006).

Pour fonctionner, cette variante de PSO retire la composante de la meilleure position globale g de l'Équation 3-1 de mise à jour de la vélocité d'une particule i. N'ayant plus de composante globale, le CLPSO se distingue des autres variantes par l'ajout d'un processus de mise à jour de la meilleure position cognitive p_i avec l'apprentissage compréhensif. Chaque fois que p_i n'est pas améliorée par la mise à jour d'une particule, un compteur m_i spécifique à cette particule i est augmenté de 1. Lorsque ce compteur atteint la valeur critique M, lequel est un nouveau paramètre fixé à 7 par Liang et al. (2006), le processus d'apprentissage compréhensif est enclenché. Afin d'obtenir la contribution collective de l'essaim, une sélection par tournoi est déclenchée comme apprentissage afin que les particules apprennent des autres. Une particule peut donc modifier la composition de sa

meilleure solution cognitive p_i en fonction de la performance des autres particules ayant gagné la sélection par tournoi.

Considérant l'absence de la composante g pour la mise à jour d'une particule, le CLPSO remplace l'Équation 3-1 provenant du PSO par l'Équation 3-9 qui suit. Le paramètre d'inertie w utilisé est celui correspondant à l'Équation 3-5 de Ratnaweera et al. (2004), c'est-à-dire celui comportant une modification non linéaire du paramètre en fonction des itérations de l'algorithme.

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (p_i - x_i^t) \tag{3-9}$$

Le fonctionnement du CLPSO est illustré à la Figure 3-4, où N représente le nombre total de particules de l'essaim. L'algorithme débute avec l'encadré 3-4A représentant le processus de mise à jour d'une particule. Au début de cette mise à jour, il est nécessaire de vérifier si la particule i à l'itération t est demeurée stagnante pendant trop de générations. La vérification de la stagnation est effectuée à l'intérieur de l'encadré 3-4C. Un processus d'apprentissage compréhensif est enclenché en (1) s'il y a eu stagnation pendant au moins M générations. Une fois cet apprentissage complété, le CLPSO poursuit son exécution par la mise à jour de la particule avant l'évaluation par la fonction objectif. La meilleure solution cognitive et la meilleure solution globale sont ensuite comparées avec la nouvelle solution produite dans l'encadré 3-4B. Cette portion de l'algorithme est très similaire à l'encadré 3-2B du PSO provenant de la Figure 3-2, à l'exception qu'il y a en plus une mise à jour du compteur m_i pour une particule i. Lorsque la meilleure position cognitive p_i d'une particule n'est pas améliorée lors d'une itération, le compteur m_i est incrémenté de 1. À l'opposé, s'il y a une amélioration, le compteur retourne à la valeur 0. Ces opérations sont répétées pour les N particules pendant les T générations.

Le processus d'apprentissage compréhensif enclenché en ① se base sur une probabilité d'apprentissage Pc_i calculée spécifiquement pour chacune des particules de l'essaim. La valeur de ce paramètre est fixée selon l'Équation 3-10, laquelle est initialisée en même temps que les autres composantes au début de l'algorithme. Cette valeur demeure constante sur l'ensemble des générations et dépend du nombre de particules utilisées afin que chaque particule n'ait pas la même probabilité d'apprentissage.

$$Pc_i = 0.05 + 0.45 \cdot \frac{\left(exp\left(\frac{10\cdot(i-1)}{N-1}\right)-1\right)}{(exp(10)-1)} \qquad (3\text{-}10)$$

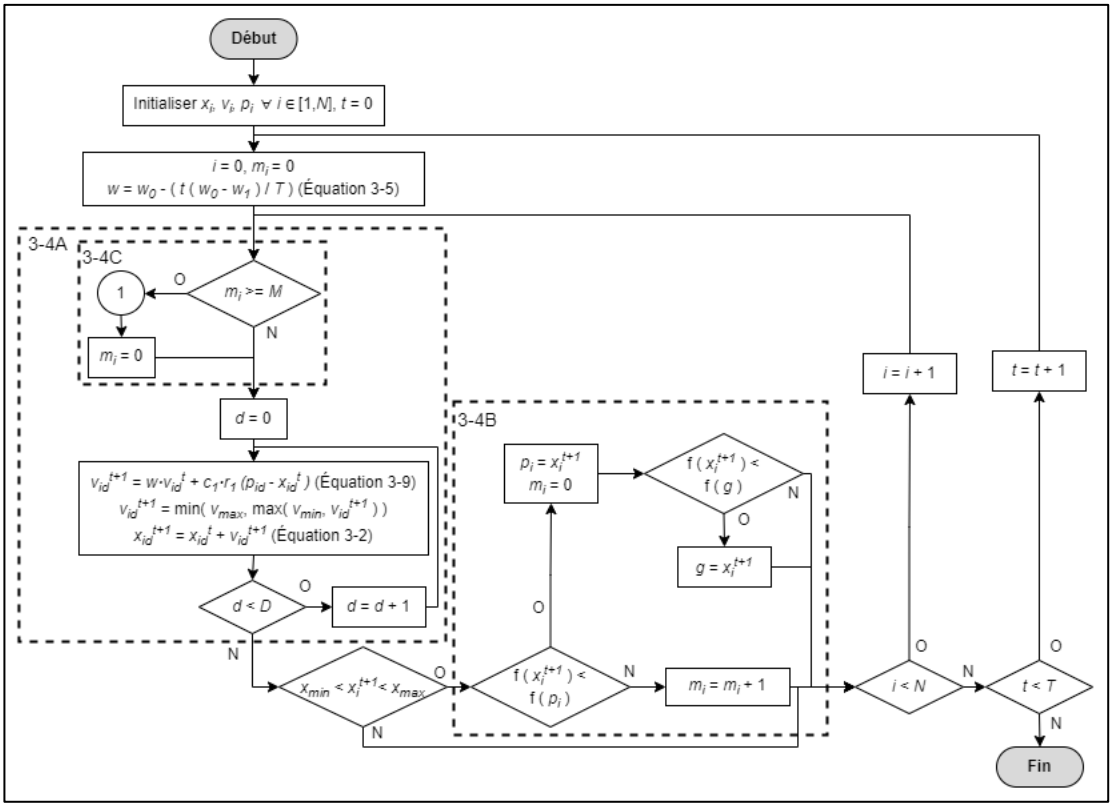

Figure 3-4 : Organigramme du fonctionnement du CLPSO. 3-4A - Mise à jour d'une particule. 3-4B - Évaluation et mise à jour des meilleures solutions (cognitive et globale). 3-4C – Vérification pour le déclenchement du processus d'apprentissage compréhensif.
© Hugo Deschênes

La Figure 3-5 illustre les opérations effectuées lorsque l'apprentissage compréhensif est enclenché pour le CLPSO. Un nombre réel aléatoire r_1 situé entre 0 et 1 est généré pour

chacune des dimensions d de la particule. Si ce nombre est inférieur à la valeur de la probabilité d'apprentissage Pc_i de la particule en traitement, deux autres particules parmi celles de l'essaim (i_1 et i_2) sont sélectionnées aléatoirement selon deux nouveaux nombres réels (r_2 et r_3) tirés entre 0 et 1 et multipliés par la grandeur N de l'essaim. La meilleure des deux particules (i_1 ou i_2) selon l'évaluation par la fonction objectif $f(.)$ de leur meilleure solution cognitive respective est utilisée afin de copier le contenu de la dimension d de cette solution cognitive dans la dimension correspondante au sein p_{id}. Les variables $p_{i_1 d}$ et $p_{i_2 d}$ représentent les valeurs contenues à la dimension d des meilleures solutions cognitives p_{i_1} et p_{i_2}. Ces solutions proviennent des particules sélectionnées aléatoirement (i_1 et i_2).

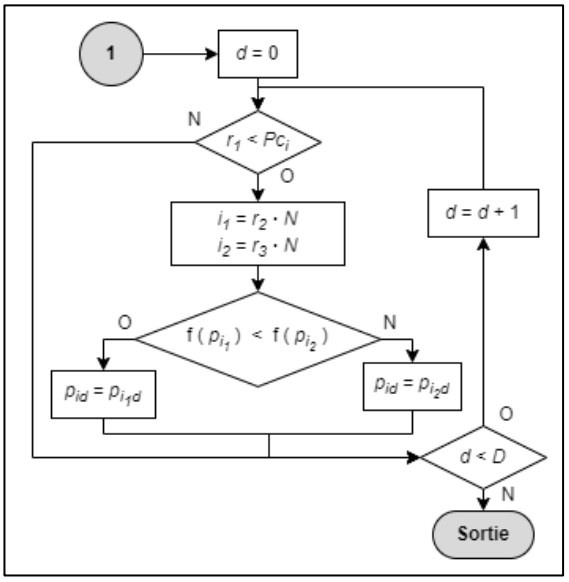

Figure 3-5 : Organigramme de la sélection par tournoi du processus d'apprentissage compréhensif du CLPSO (Liang et al., 2006)
© Hugo Deschênes

Une particularité des auteurs avec le CLPSO est que si la position générée par une particule sort de l'espace de recherche après la mise à jour de ses D dimensions, elle n'est pas contrainte et déplacée sur la limite de l'espace du problème. Son évaluation sera plutôt ignorée considérant qu'à l'itération suivante, la particule reviendra naturellement à l'intérieur

de l'espace de recherche grâce à l'influence des autres particules et de sa mémoire personnelle.

Le CLPSO est entre autres utilisé dans la littérature pour la résolution du problème de répartition de la puissance réactive (*reactive power dispatch problem*) (Mahadevan & Kannan, 2010), de même que pour la résolution du problème d'optimisation contraint avec variables mixtes (*constrained mixed variable optimization problem*) (Gao & Hailu, 2010). Un autre problème abordé avec cette méthode est celui de la reconnaissance d'images selon la perception des couleurs visibles par l'humain (Puranik et al., 2009). Tout comme le BPSO, le CLPSO a lui aussi été utilisé dans la littérature pour diverses améliorations. Des auteurs ont d'ailleurs utilisé un concept d'opposition de solutions (W. Wang et al., 2011; Z. Wu et al., 2008). Ce concept s'est avéré utile pour l'optimisation avec l'algorithme d'évolution différentielle (Storn & Price, 1997).

3.2.3.3. *COOPERATIVE LEARNING* PSO (CoLPSO)

Le CoLPSO est une variante de PSO conçue dans le but de pallier la difficulté du PSO à traiter les problèmes de nature continue de grande taille (van den Bergh & Engelbrecht, 2004). Les auteurs de cette variante indiquent que la performance du PSO décline lorsque la taille du problème augmente. Le CoLPSO contourne cette faiblesse en optimisant séparément chacune des dimensions (variables de décision) du problème étudié.

Afin d'y parvenir, la boucle itérant sur les dimensions de l'algorithme du PSO est échangée avec la boucle itérant sur les particules de l'essaim. Le CoLPSO fonctionne en sens inverse, où chaque dimension de la solution possède sa propre population, c'est-à-dire, son propre sous-essaim de N particules. Au lieu de mettre à jour les D dimensions pour chacune des N particules les unes après les autres, l'algorithme mets à jour N particules unidimensionnelles pour chacune des D dimensions, une dimension à la fois. Il n'y a donc

pas utilisation de N particules comme dans le PSO, mais plutôt $D\ x\ N$ particules. Ceci permet à l'algorithme de mettre à jour chaque dimension en diminuant l'influence qu'elle subit par les les changements simultanés apportés à une autre dimension provenant de la même solution. Ceci consiste à partitionner une solution.

Le partitionnement d'une solution est utilisé afin d'isoler chacune des variables de décision (dimension) et pour évaluer leur contribution individuelle à la fonction objectif. L'essaim de particules du CoLPSO est composé d'un nombre de partitions, aussi nommés sous-essaims, égal à la dimension D du problème. Chaque sous-essaim est composé de N particules de grandeur 1, formant une population générale de $D\ x\ N$ particules. Chaque sous-essaim correspond à une dimension d d'une solution et comporte chacun une meilleure solution globale g_d.

Tel qu'expliqué précédemment, la conception du CoLPSO vise à réduire au minimum l'impact du changement des dimensions sur les autres variables de décision. Avec le partitionnement de l'espace de recherche en sous-essaims, des pseudo-minimums peuvent être créés (van den Bergh & Engelbrecht, 2004). Ceci implique que la fixation de certaines valeurs pour une dimension donnée peut limiter les valeurs possibles pour une autre dimension, créant ainsi des minimums qui n'existent pas lorsque les dimensions d'une solution sont mises à jour simultanément. La valeur de la fonction objectif lors de l'évaluation du vecteur contextuel est donc limitée par la fixation des valeurs pour les autres dimensions.

Afin de permettre une évaluation individuelle pour chacune des variables de décision, le CoLPSO utilise un vecteur contextuel nommé cv de grandeur identique à la dimension D. Ce vecteur est constitué d'une copie de chaque meilleure solution g_d provenant de chacun des sous-essaims. Il est le seul médium par lequel les sous-essaims peuvent communiquer entre eux leurs solutions. L'évaluation d'une particule appartenant à une dimension d s'effectue en remplaçant la valeur de sa position actuelle à l'emplacement d correspondante

au sein de cv. Le vecteur cv est ensuite évalué avec la fonction objectif afin de mesurer la qualité de ce changement. Si la solution s'améliore, cv conserve la valeur testée à l'emplacement d et elle est ensuite enregistrée dans g_d. Si la solution ne s'améliore pas, la valeur contenue à l'emplacement d de cv est ignorée et elle retrouve sa valeur initiale avant l'évaluation. Le vecteur contextuel est donc la composante permettant de communiquer de l'information entre les différents sous-essaims.

La Figure 3-6 suivante illustre la structure organique du PSO (à gauche) en comparaison avec le CoLPSO (à droite), où chaque cercle représente une particule et un ovale représente un essaim. Dans l'exemple pour le PSO, une dimension $D = 5$ et une population $N = 10$ engendre un ensemble de 10 particules comportant chacune une position et une vélocité de grandeur 5. Pour le CoLPSO, une dimension $D = 5$ et une population $N = 10$ engendre 5 sous-essaims comportant chacun 10 particules pour un total de 50 particules unidimensionnelles. Chacune des particules contient une position et une vélocité de grandeur 1 et est associée à une des dimensions du problème. Également, le PSO ne contient qu'une seule composante de meilleure solution globale g, alors que le CoLPSO en contient autant qu'il y a de dimension, c'est-à-dire, 5. Ces composantes g pour le CoLPSO forment le vecteur contextuel cv.

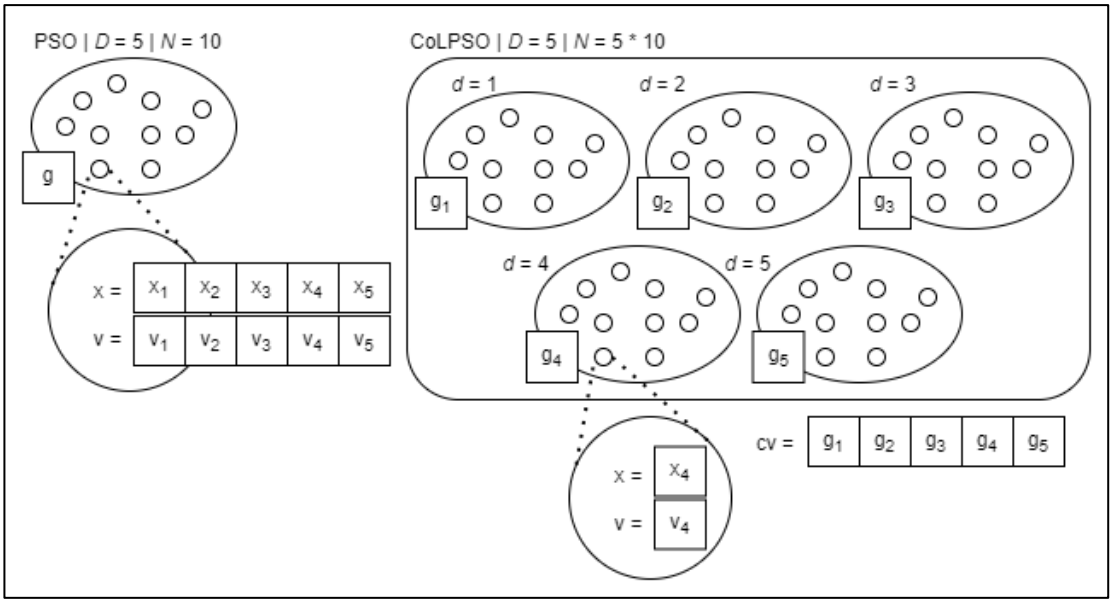

Figure 3-6 : Illustration de la structure organique du PSO en comparaison avec le CoLPSO, pour un exemple de problème comportant une dimension définie à 5 et une population de 10 particules.
© Hugo Deschênes

La Figure 3-7 illustre le fonctionnement du CoLPSO. Dans les variantes précédentes, la mise à jour d'une particule utilise la lettre A pour la notation des encadrés des organigrammes. Considérant qu'une particule pour le CoLPSO ne se met pas à jour selon les mêmes principes que ces variantes, la notation D est plutôt utilisée ici pour la mise à jour d'une dimension. Ainsi, l'encadré 3-7D regroupe le processus de mise à jour d'un sous-essaim. Il débute avec la mise à jour de la position et de la vélocité d'une particule i selon l'Équation 3-1 et l'Équation 3-2 provenant du PSO. La nouvelle position générée x_{id}^{t+1} est ensuite copiée à la dimension d du vecteur contextuel cv. Lorsqu'un sous-essaim est mis à jour, la valeur contenue à l'intérieur des particules est copiée dans la dimension correspondante du vecteur contextuel. Ce vecteur est ensuite évalué à l'aide de la fonction objectif $f(.)$ tel qu'illustré au début de l'encadré 3-7B. Si cette nouvelle valeur améliore la solution produite par cv, la position de la particule remplace la meilleure solution cognitive p_{id} de la particule i. La meilleure solution globale du sous-essaim g_d est également mise à jour si cette solution permet de l'améliorer. Une fois les N particules traitées, la meilleure valeur obtenue par le sous-essaim est copiée à l'intérieur de cv en vue d'être utilisée par les autres

sous-essaims du CoLPSO. Ce processus est répété pour tous les sous-essaims pendant T générations.

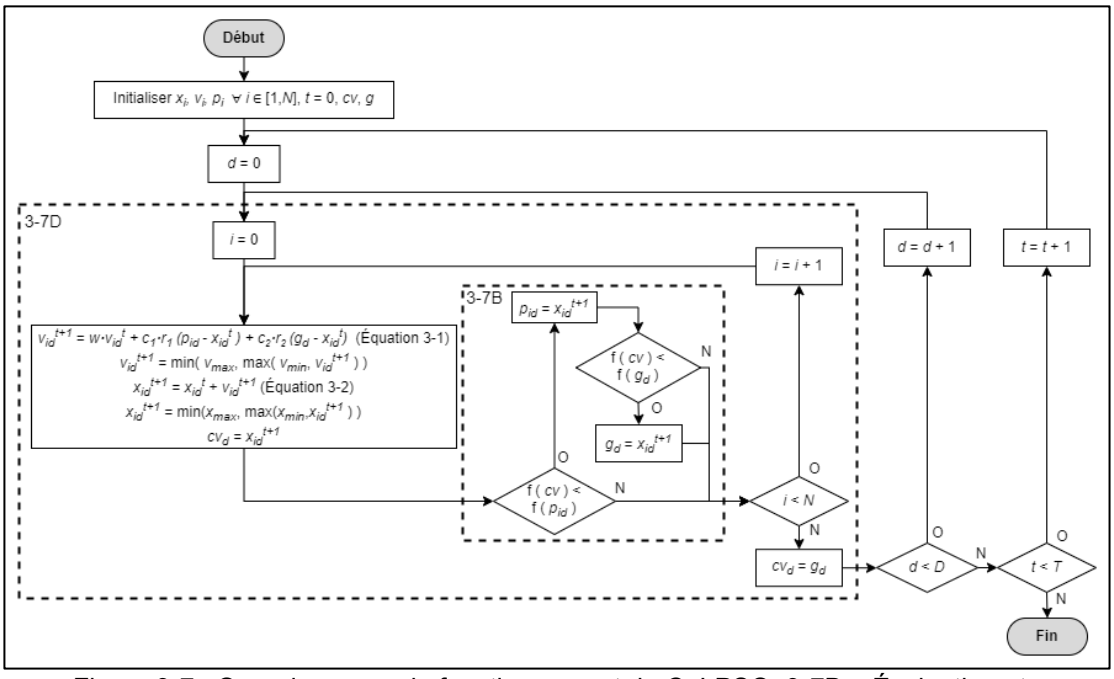

Figure 3-7 : Organigramme du fonctionnement du CoLPSO. 3-7B – Évaluation et mise à jour des meilleures solutions (cognitive et globale). 3-7D – Mise à jour d'un sous-essaim.
© Hugo Deschênes

Une version alternative au CoLPSO est proposée dans la littérature pour la résolution de problèmes multimodaux de nature continue (J. Zhang & Ding, 2011). Les auteurs de cette version font usage de quatre instances de CoLPSO exécutées en parallèle, en modifiant certains paramètres ou certaines topologies de communication inter-particules. Un autre exemple pour la résolution des mêmes types de problèmes est proposé par X. Li et Yao (2012). Ces derniers effectuent plutôt une hybridation entre le CoLPSO, le CLPSO et le BPSO. Un hybride entre le CoLPSO et le CLPSO est également proposé par Maitra et Chatterjee (2008) afin de résoudre le problème de partitionnement d'images.

3.3. HYBRIDATION AVEC LE PSO

Le PSO est une métaheuristique très utilisée dans la littérature afin de contribuer à améliorer les solutions par l'hybridation (Hussain et al., 2019; Sengupta et al., 2018). Plusieurs de ces hybridations ont comme principal objectif de pallier les lacunes qu'a le PSO à explorer efficacement l'espace de recherche lors de la résolution de grandes instances de problème (Tamayo-Vera et al., 2018; van den Bergh & Engelbrecht, 2004). Les hybridations de PSO décrites dans cette section sont divisées selon la nature *homogène/hétérogène* de l'hybride provenant de la classification selon Talbi décrite à la Section 2.3.2.

3.3.1. HYBRIDATIONS DE NATURE *HÉTÉROGÈNE*

Tel que stipulé par Talbi, l'hybridation *hétérogène* consiste à regrouper en un seul processus au moins deux méthodes de résolution de nature différente. Il peut s'agir d'une méthode de résolution exacte avec une métaheuristique, ou encore au moins deux métaheuristiques de différente nature.

Plusieurs exemples sont recensés dans la littérature approuvant les bienfaits et les avantages d'utiliser le PSO lors d'hybridations hétérogènes. Par exemple, le PSO est utile afin d'aider à fixer les paramètres d'un modèle de Markov caché (J. Sun et al., 2014; J. Sun et al., 2012) et pour configurer un réseau de neurones artificiels (Y. Sun et al., 2011; Xiong et al., 2015). Il est également hybridé avec l'algorithme de recherche avec tabous afin de résoudre le problème d'ateliers à cheminements multiples (*job shop scheduling problem*) (Sha & Hsu, 2006), avec une recherche locale pour la résolution du problème d'ordonnancement séquentiel (*sequential ordering problem*) (Anghinolfi et al., 2011) et avec un algorithme d'optimisation par colonies de fourmis pour la résolution de problèmes de nature continue (Shelokar et al., 2007). Plusieurs hybridations, entre l'algorithme génétique et le PSO, sont également proposées dans la littérature (Banks et al., 2008; Soleimani &

Kannan, 2015; J. Wu et al., 2015). Certaines composantes de l'algorithme génétique, telles que la mutation, le processus de sélection ou encore le croisement, sont ajoutés au PSO par certains auteurs afin de résoudre des fonctions continues et des problèmes combinatoires tels que le problème du voyageur de commerce. Un article plus récent démontre qu'une hybridation entre le PSO et le troupeau de krills (*krill herd algorithm*) permet d'obtenir des résultats intéressants sur un grand nombre de fonctions continues, où la moitié de la population évolue selon le PSO et l'autre moitié selon le troupeau de krills (G.-G. Wang et al., 2016). Une autre méthode hybride utilise une combinaison du PSO avec l'algorithme de recherche à voisinage variable (*variable neighborhood search*) afin de trouver le plus court chemin sous contraintes. Il y a également un article publié par Mondal et al. (2018) proposant une hybridation entre le PSO et l'algorithme des colonies d'abeilles pour résoudre le problème d'amélioration du contraste d'images (*image contrast enhancement problem*). Finalement, une hybridation entre l'évolution différentielle et le PSO est proposée par Du et Liu (2020) pour la résolution de fonctions continues.

3.3.2. HYBRIDATIONS DE NATURE *HOMOGÈNE*

Utiliser deux métaheuristiques organiquement différentes n'est pas le seul moyen de produire un nouvel hybride. Un autre moyen permettant d'atteindre ce but est de combiner différentes variantes d'un même algorithme afin de pallier les faiblesses d'une des variantes avec les forces de l'autre, et vice-versa, sans qu'il soit nécessaire de changer la métaheuristique utilisée. Il s'agit d'une hybridation *homogène*.

Plusieurs hybridations *homogènes* sont proposées dans la littérature par l'utilisation des variantes décrites à la Section 3.2.3. Le BPSO en est un exemple lorsqu'il est hybridé avec le CoLPSO en permettant une génération aléatoire de valeurs lors de la mise à jour d'une particule unidimensionnelle (B. Jiang & Wang, 2014; van den Bergh & Engelbrecht, 2004). Une autre hybridation, composée du CLPSO et du CoLPSO, est proposée pour la

résolution du problème de segmentation d'images (Maitra & Chatterjee, 2008). Dans cet algorithme, le processus d'apprentissage compréhensif du CLPSO est ajouté au début de la mise à jour d'une particule du CoLPSO. Un autre hybride utilise quatre variantes de PSO exécutées séparément pour la résolution d'un même problème (G.-W. Zhang et al., 2015). Cet algorithme utilise le calcul haute performance afin d'apporter une communication inter-PSO à des intervalles réguliers tout au long de son exécution. Le but recherché par cet hybride est d'ajouter l'échange des meilleures solutions globales provenant de chacune des variantes à des intervalles réguliers. Ces hybridations apportent différentes manières de modéliser les interactions possibles entre des variantes de PSO, par l'ajout de composantes, l'exécution simultanée de variantes et le changement de comportement des particules.

La prochaine section porte sur la deuxième stratégie permettant d'améliorer les métaheuristiques : la discrétisation. Elle reprend les concepts vus à la Section 2.4 avec une approche axée sur l'utilisation du PSO.

3.4. DISCRÉTISATION DU PSO

Les particules du PSO sont conçues afin d'effectuer des déplacements dans un espace de recherche composé de nombres réels. L'Équation 3-1 de mise à jour de la vélocité d'une particule et l'Équation 3-2 de mise à jour de sa position utilisent les opérateurs arithmétiques d'addition, de soustraction et de multiplication. Ces opérateurs ne peuvent pas être utilisés tels que définis lorsque la nature des variables change, donc lorsqu'elles ne sont plus codifiées en valeurs réelles. Il est nécessaire de redéfinir leur comportement afin d'adapter le déplacement d'une particule en fonction des variables de décision lorsque le PSO est utilisé pour l'optimisation d'un problème discret ou binaire.

Afin d'établir une marche à suivre dans le but de discrétiser le PSO, Clerc (2004) propose des éléments à redéfinir permettant de conserver le fonctionnement général de la

métaheuristique. Ces redéfinitions assurent que les équations régissant le comportement du PSO prennent en considération la nouvelle codification des variables composant une solution. Les éléments à spécifier sont :

- La nature des valeurs composant la position d'une particule ;
- La nature des valeurs composant la vélocité d'une particule ;
- Le résultat de l'addition entre une position et une vélocité (déplacement) ;
- Le résultat de la soustraction entre deux positions (distance) ;
- Le résultat de l'addition entre deux vélocités ; et
- Le résultat de la multiplication entre un nombre réel et une vélocité.

Redéfinir ces éléments permet de s'assurer que tous les opérateurs arithmétiques composant l'Équation 3-1 et l'Équation 3-2 sont utilisables afin de mettre à jour les particules de l'essaim.

Tel que mentionné à la Section 2.4.3, Krause et al. (2013) font état de six procédés de discrétisation applicables aux métaheuristiques continues à base d'intelligence en essaims. Le sixième des procédés est consacré à la discrétisation du PSO. La littérature fait mention de différents procédés de discrétisation permettant d'adapter le PSO pour la résolution de problèmes discrets. Le Tableau 3-1 recense plusieurs articles provenant de la littérature sur la discrétisation du PSO en les répartissant en six catégories régissant leur fonctionnement principal : l'arrondi, la manipulation de nombres réels, l'utilisation d'une composante binaire, l'utilisation d'une composante ternaire, la manipulation de listes de nombres entiers et l'utilisation d'une fonction de construction et de destruction de solutions. Pour chacune de ces catégories, une courte description du procédé de discrétisation est spécifiée, de même que la forme utilisée pour la codification de la position x (*Pos*) et de la vélocité v (*Vél*) de la particule (E-Entière, R-Réelle, B-Binaire, T-Ternaire et -- Aucun). Les articles utilisant ces procédés sont également listés dans la dernière colonne.

Tableau 3-1 : Résumé de plusieurs procédés de discrétisation appliqués au PSO.

PROCÉDÉ	POS	VÉL	ARTICLES
Arrondi			
Troncature des décimales après chaque calcul.	E	E	(Shayeghi et al., 2010a) (Shayeghi et al., 2010b) (Jin et al., 2007) (Yusoff et al., 2015) (Jia & Seo, 2013) (T. Chen et al., 2006) (Salman et al., 2002)
Arrondissement des valeurs lors de la mise à jour de la position d'une particule.	E	R	(Del Valle et al., 2008) (Ziari et al., 2011) (Laskari et al., 2002)
Manipulation de nombres réels			
Utilisation d'un vecteur ou d'une matrice de priorités pour la composition de la position et/ou de la vélocité.	R	R	(R.-M. Chen et al., 2010) (H. Zhang et al., 2005) (H. Zhang et al., 2006) (Pang et al., 2004) (Sha & Hsu, 2006) (Akjiratikarl et al., 2007) (Ali et al., 2020)
Heuristique de transformation de solutions lors de la mise à jour des particules.	E	R	(Cuevas et al., 2011) (Onwubolu & Davendra, 2009)
Utilisation de listes de nombres réels représentant des points de coupe à effectuer au sein de la composition d'une position.	R	R	(Strnad & Kohek, 2017)
Avec composante binaire			
Transformation logistique de la vélocité.	B	R	(J. Kennedy & Eberhart, 1997) (Miao et al., 2021)
Utilisation de traces pour la vélocité.	B	R	(Liao et al., 2007)
Mutation des particules selon une ondelette	B	R	(F. Jiang et al., 2017)
Permutations d'éléments avec utilisation d'une liste de tabous.	E	B	(Sha & Hsu, 2006)
Avec composante ternaire			
Vélocité sélective de la direction et permutations d'éléments.	E	T	(Tian et al., 2013)
Avec vecteur transitoire ternaire ajouté à la formulation du PSO.	E	R	(B. Jarboui et al., 2008) (Bassem Jarboui et al., 2008)
Manipulation de listes de nombres entiers			
Liste de translations d'éléments à appliquer à une position.	E	E	(Anghinolfi & Paolucci, 2009) (Anghinolfi et al., 2011) (Lemamou et al., 2010) (X. Wang et al., 2013)

PROCÉDÉ	POS	VÉL	ARTICLES
Liste de permutations d'éléments à appliquer à une position.	E	E	(Clerc, 2004) (Ponnambalam et al., 2009) (Zhong & Lei, 2010) (X. H. Shi et al., 2007)
Translations aléatoires d'éléments	E	E	(Ling Wang et al., 2010)
Utilisation d'opérateurs de mutation	E	--	(Liu et al., 2020) (Liu et al., 2021)
Construction/Destruction de solutions			
Utilisation d'un algorithme glouton itératif et d'un croisement entre deux solutions.	E	--	(Tasgetiren et al., 2007) (Pan et al., 2008)

Tel que le montre le Tableau 3-1, une grande variété de procédés de discrétisation sont proposés dans la littérature. Cependant, trois de ces procédés ressortent davantage étant donné leur conception et leur popularité. Ils sont décrits en détails à la Section 5.3 et sont considérés dans le cadre de cette thèse. Les procédés concernés sont :

- Par manipulation de nombres réels selon H. Zhang et al. (2005);
- Par manipulation de nombres entiers selon Anghinolfi et Paolucci (2009);
- Par manipulation de nombres entiers selon Clerc (2004).

La prochaine section présente différents constats reliés à l'utilisation du PSO lors de l'hybridation et de la discrétisation de métaheuristiques pour la résolution de grandes instances de problèmes. Elle présente ensuite les divers objectifs de la recherche fixés dans le cadre de cette thèse, ainsi que la méthodologie utilisée afin de les accomplir.

3.5. PROBLÉMATIQUE ET OBJECTIFS DE LA RECHERCHE

Plusieurs métaheuristiques ont été élaborées à ce jour afin d'obtenir de bonnes solutions pour divers problèmes d'optimisation, qu'ils soient continus ou combinatoires. Ces

algorithmes viennent pallier les limitations qu'ont les méthodes exactes à résoudre des problèmes NP-difficiles. Les métaheuristiques ont été élaborées afin d'obtenir une solution de bonne qualité en un temps raisonnable sur ce type de problème. Elles sont régulièrement utilisées afin de répondre aux besoins des entreprises (Boussaïd et al., 2013). Certaines métaheuristiques, telles que l'algorithme génétique (Goldberg & Holland, 1988), l'optimisation par colonies de fourmis (Dorigo & Gambardella, 1997) ou encore la recherche avec tabous (Glover, 1989, 1990), ont été conçues pour la résolution de problèmes discrets. D'autres ont plutôt été conçues pour la résolution de problèmes continus, telles que l'optimisation par essaims particulaires (PSO) (R. Eberhart & Kennedy, 1995) et l'évolution différentielle (Storn & Price, 1997).

Tel que stipule le théorème du « No Free Lunch », l'utilisation d'une seule méthode ne peut être capable à elle seule de résoudre efficacement la multitude de problèmes auxquels font face les industries (Wolpert & Macready, 1997). Ce théorème précise que si un algorithme résout efficacement un problème en particulier, il sera inévitablement moins efficient pour plusieurs autres. Ceci implique également qu'il y aura toujours de nouvelles approches permettant d'améliorer les algorithmes d'optimisation afin de contribuer efficacement à la résolution de problèmes précis (Weise et al., 2009). Fister Jr et al. (2013), de même que Hussain et al. (2019), ont recensé une grande partie des nouvelles métaheuristiques ayant été élaborées depuis quelques années. Étant donné la grande quantité proposée dans la littérature, il peut s'avérer difficile de cerner efficacement quelle méthode performe mieux que les autres et sur quels problèmes elles sont efficaces. Une métaheuristiques doit être adaptée à un problème pour qu'elle soit efficace (El-Ghazali Talbi, 2009). En effet, la qualité des résultats de la résolution d'un problème est fortement tributaire de la qualité de l'adaptation de la métaheuristique à celui-ci. Selon Hussain et al. (2019), la recherche devrait être davantage orientée afin d'améliorer les métaheuristiques existantes et les mécaniques régissant leur comportement que sur la création de nouvelles méthodes. Une des avenues de recherche prometteuses consiste alors à proposer une amélioration des

méthodes existantes par l'hybridation (Boussaïd et al., 2013; Sengupta et al., 2018) et par la discrétisation (Abdel-Basset et al., 2018; Krause et al., 2013).

Parmi la variété de métaheuristiques de la littérature, on retrouve des méthodes à solution unique et des méthodes à base de population. Parmi les algorithmes à base de population, on distingue deux catégories : les algorithmes évolutionnaires et les algorithmes d'intelligence en essaims. L'optimisation par essaims particulaires fait partie de la branche des algorithmes d'intelligence en essaims. L'avantage principal des méthodes de résolution appartenant à cette branche est leur flexibilité et leur robustesse (Lalwani et al., 2015). Les possibilités d'adaptation de cette structure comportementale, qu'est l'interaction à l'intérieur d'une population, permettent une bonne variété d'implémentations et d'adaptations. Ce type de métaheuristiques offre ainsi une très grande flexibilité pour leur adaptation à des problèmes de diverses natures. La présence d'une population offre la possibilité de définir plusieurs comportements différents régissant l'interaction entre les individus. Elles offrent en même temps la possibilité de transposer ces comportements d'une méthode à une autre méthode basée également sur l'évolution d'une population.

Lors de la résolution de problèmes combinatoires, une augmentation de la taille du problème cause une explosion de la quantité de solutions potentielles. Généralement, la performance de certaines métaheuristiques se détériore rapidement lorsque la dimension d'un problème augmente, diminuant ainsi la scalabilité de ces métaheuristiques (Boussaïd et al., 2013). L'ajustement entre l'exploration et l'exploitation de l'espace de recherche devient de plus en plus difficile et d'autant plus important. L'objectif principal de cette thèse est donc de *Contribuer aux stratégies métaheuristiques à base de population pour résoudre efficacement des problèmes de grande taille*. La réalisation de cet objectif principal est rendue possible grâce à l'atteinte de plusieurs objectifs secondaires touchant l'hybridation, la discrétisation et la considération des instances de grande taille dans la conception de stratégies métaheuristiques. Ces objectifs secondaires sont formulés de la manière suivante :

- *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature* ;
- *Adapter et comparer trois méthodes de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support* ; et
- *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA.*

3.5.1. PREMIER OBJECTIF SECONDAIRE

Le premier objectif secondaire est de *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature*. Un moyen permettant d'améliorer les performances d'une méthode d'optimisation consiste à l'hybrider avec une autre méthode (Blum et al., 2011; Sengupta et al., 2018). L'hybridation sert principalement à combiner les forces de chacune des méthodes et à réduire par le fait même l'impact de leurs lacunes. Améliorer l'exploration de l'espace de recherche et l'exploitation des solutions est donc le but principal visé par l'hybridation. Là ou une méthode de résolution propose de moins bonnes performances, une autre méthode de résolution peut offrir de meilleurs résultats et coopérer à l'obtention d'une meilleure solution résultante. L'hybridation peut aider les métaheuristiques à obtenir une meilleure solution en comparaison de l'utilisation individuelle d'une méthode sur les mêmes instances de problème. Plusieurs formes d'hybridation se retrouvent dans la littérature, lesquelles ont été recensées par E-G Talbi (2002), Jourdan et al. (2009), de même que Raidl (2006). Les possibilités d'hybridation sont très grandes et ne peuvent toutes être couvertes dans un même ouvrage. Dans le cadre de cette thèse, trois formes d'hybridations homogènes sont testées dans le but d'améliorer la flexibilité du PSO. Les améliorations permettent de transposer les stratégies d'hybridation utilisées pour l'application sur une plus grande variété de métaheuristiques à base de population. Plusieurs auteurs ont d'ailleurs contribué grandement à l'amélioration du PSO en proposant différentes hybridations de cette

métaheuristique, mettant à jour les solutions de la population de différentes manières (Anghinolfi et al., 2011; Banks et al., 2008; Mondal et al., 2018; Sha & Hsu, 2006).

Le choix du PSO en tant que méthode d'optimisation s'explique par les bonnes performances dont elle a fait preuve dans la littérature (Banks et al., 2008). Cette méthode est également sélectionnée grâce à sa nature purement continue et par les possibilités d'adaptation qu'elle offre pour la résolution de problèmes discrets (Krause et al., 2013). Considérant que le PSO est une métaheuristique grandement utilisée en hybridation (Hussain et al., 2019), l'atteinte de ce premier objectif secondaire débute par la sélection de trois variantes de PSO permettant chacune d'améliorer la méthode originale. Cette sélection s'effectue également selon la nature différente des variantes et leur structure potentiellement complémentaire. La première variante, le BPSO (James Kennedy, 2003), apporte un déplacement pseudo-aléatoire des particules selon une loi normale de la distribution des solutions lors de la mise à jour d'une particule. La deuxième variante, le CLPSO (Liang et al., 2006), ajoute un processus d'apprentissage compréhensif applicable à chaque particule n'ayant pas amélioré la meilleure solution collective pendant un trop grand nombre de générations. La troisième variante, le CoLPSO (van den Bergh & Engelbrecht, 2004), résout les problèmes en traitant et en évaluant séparément chaque dimension de la solution. Par l'utilisation d'hybridations de nature *homogène* (E-G Talbi, 2002), trois combinaisons sont formées entre les variantes mentionnées ci-haut afin de compenser les faiblesses d'une des variantes par les forces d'une autre. Les trois variantes et les trois hybridations sont implémentées en C++. Les nouveaux hybrides sont testés sur des fonctions classiques en optimisation continue à dimensions variables, soient *Sphere*, *Ackley*, *Rosenbrock*, *Rastrigin*, *Schwefel* et *Griewank* (Dixon & Szego, 1978; Molga & Smutnicki, 2005). Ces expérimentations permettent de cibler les forces de chacun, leurs faiblesses, leur applicabilité et leur capacité à résoudre efficacement des problèmes de grandeur et de complexité variable. La meilleure des hybridations formées est également testée sur des fonctions plus récentes et plus complexes avec l'ensemble de référence *CEC'15* établi par Qu et al. (2016).

Elle est comparée avec un algorithme de la littérature nommé dynFWACM (Yu et al., 2015). La comparaison des résultats est réalisée selon les valeurs maximales et minimales obtenues, de même que les moyennes, médianes et déviations standards sur 50 exécutions avec l'ensemble des fonctions tests. La performance de la méthode hybride sélectionnée permet de valider la contribution positive de l'hybridation de variantes de PSO pour la résolution de problèmes de nature continue.

3.5.2. DEUXIÈME OBJECTIF SECONDAIRE

Le deuxième objectif secondaire est d'*Adapter et de comparer trois procédés de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support*. Parmi la diversité des métaheuristiques à base de population dans la littérature, le PSO figure parmi les méthodes les plus performantes pour la résolution de problèmes à variables réelles. Elle offre une bonne flexibilité et nécessite l'utilisation d'opérateurs arithmétiques, ce qui en fait une méthode plus facilement modifiable et plus accessible. Le PSO a été démontré performant pour la résolution d'une variété de problèmes provenant de différents champs d'intérêt, tels que l'optimisation de fonctions mathématiques (Banks et al., 2008; James Kennedy, 2003; Liang et al., 2006; Qu et al., 2016), la résolution de l'ordonnancement séquentiel (Anghinolfi et al., 2011), la paramétrisation de réseaux de neurones (Y. Sun et al., 2011), la paramétrisation d'un modèle de Markov caché (HMM) (J. Sun et al., 2012) et l'alignement multiple de séquences (Lalwani et al., 2015). Par contre, sa performance pour la résolution de problèmes combinatoires est variable selon différents facteurs, dont la dimension de la solution d'une particule et le niveau de contraintes du problème (Jordehi & Jasni, 2015; van den Bergh & Engelbrecht, 2004). De plus, une grande majorité des problèmes résolus à l'aide du PSO sont de nature continue (Hussain et al., 2019). La discrétisation du PSO apporte une flexibilité intéressante à son utilisation en permettant la résolution de problèmes à variables entières. Plusieurs méthodes de discrétisation existent déjà dans la littérature (Jordehi & Jasni, 2015). Elles sont cependant

difficiles à comparer, car elles ont souvent été expérimentées pour la plupart dans des conditions différentes, pour des problèmes différents.

Le PSO est encore une fois utilisé afin de valider cet objectif secondaire. Afin d'adapter et de comparer la discrétisation avec le PSO, trois différents procédés de discrétisation parmi ceux listés à la Section 2.4.3 sont sélectionnés et testés sur deux problèmes discrets offrant des niveaux différents de contraintes. Ces procédés de discrétisation sont celui de H. Zhang et al. (2005), celui d'Anghinolfi et Paolucci (2009) et celui de Clerc (2004). Le premier problème de support, le problème d'ordonnancement séquentiel (*sequential ordering problem*), utilise des contraintes de préséances (Escudero, 1988). Le deuxième problème de support est celui de l'ordonnancement de projets avec contraintes de ressources (*ressource constrained project scheduling problem*) (Kolisch & Sprecher, 1997). Ce problème comporte des préséances entre les activités du projet, une durée de traitement et une quantité de ressources nécessaires à la réalisation de chaque activité. Au niveau plus technique, le langage de programmation C++ est utilisé afin d'implémenter les algorithmes en se basant sur Metlib, une bibliothèque de code pour la résolution de problèmes avec les métaheuristiques pour usage interne par le groupe de recherche en informatique (GRI) de l'Université du Québec à Chicoutimi. Un projet est créé pour chaque combinaison méthode-problème puis exécuté grâce à un fichier *batch* en série pour 5 exécutions par combinaison. Les résultats sont testés et comparés selon les moyennes des déviations aux meilleures solutions connues pour les deux problèmes de support. Les paramètres utilisés pour le PSO sont les mêmes que ceux utilisés par les auteurs des procédés de discrétisation. Grâce à l'utilisation des mêmes problèmes de support, il devient possible de comparer les procédés de discrétisation sur une base commune en démontrant la contribution de chaque procédé de discrétisation lors de la résolution de problèmes discrets.

3.5.3. TROISIÈME OBJECTIF SECONDAIRE

Le troisième objectif secondaire est de *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA*. Le PSO étant une métaheuristique ayant parfois de la difficulté à résoudre efficacement des problèmes combinatoires de grande taille (Jordehi & Jasni, 2015; Tamayo-Vera et al., 2018; van den Bergh & Engelbrecht, 2004; Xu & Chen, 2009), il est possible de profiter de l'hybridation et de la discrétisation afin d'améliorer la qualité des solutions obtenues. En jumelant les contributions provenant des deux premiers objectifs secondaires et en améliorant la structure en essaims de la métaheuristique, le PSO peut être adapté et amélioré pour la résolution efficace du problème de MSA. Le PSO est déjà utilisé pour résoudre ce problème (Lalwani et al., 2015; Mohamed & Aboul Ella, 2017) et il y a encore place à l'amélioration considérant qu'un alignement multiple est rarement parfait à 100%. Deux éléments sont visés par cet objectif : adapter le PSO en proposant une modélisation de la solution sous la forme d'un anneau pour la résolution du MSA, de même que proposer un nouveau comportement entre les particules pour les améliorer constamment et éviter la stagnation des solutions. Cet objectif vise donc d'améliorer les capacités du PSO pour la résolution de problèmes tels que le MSA, de même que permettre la transposition de ces contributions sur d'autres algorithmes à base de population.

Les métaheuristiques nécessitent d'être adaptées afin de résoudre efficacement un problème. Les résultats de l'hybridation et de la discrétisation des deux objectifs secondaires précédents sont nécessaires à l'accomplissement de ce troisième objectif secondaire. Afin de valider la contribution aux grandes instances de problèmes, un problème de support proposant un grand niveau de complexité est utilisé : le problème d'alignement multiple de séquences (MSA). La première étape afin de résoudre efficacement ce problème est de définir le procédé de discrétisation et le modèle d'hybridation à utiliser. Pour ce faire, la discrétisation combine les résultats provenant du deuxième objectif secondaire avec la

proposition d'une nouvelle codification de solutions : la modélisation de l'espace de recherche en anneau. Le modèle d'hybridation utilise également les résultats provenant du premier objectif secondaire afin de trouver le moyen permettant de répondre aux défis reliés à la résolution du MSA. Plusieurs modèles d'hybridation sont proposés et testés dans le but d'aider le PSO à résoudre de grandes instances de MSA. Les contributions au PSO sur le MSA sont testées en utilisant le langage de programmation C++ sur les instances comportant une dimension plus petite que 1500 provenant de l'ensemble de référence BAliBASE 3.0 (Thompson et al., 2005). L'utilisation des valeurs minimales et maximales, de même que les médianes, les moyennes et les temps d'exécution, sont utilisés afin de valider les performances sur 50 exécutions par instance. Les résultats sont comparés avec deux outils reconnus dans le domaine de la bio-informatique et utilisant chacun une approche différente pour la résolution du MSA :

- L'approche *progressive* (Notredame, 2004) avec CLUSTAL X (Larkin et al., 2007);
- L'approche *itérative* (Notredame, 2004) avec MAFFT (Edgar, 2004a, 2004b)

L'atteinte de ces trois objectifs secondaires permet de valider la contribution aux stratégies métaheuristiques par la discrétisation et l'hybridation homogène de métaheuristiques. L'utilisation du PSO permet d'améliorer les métaheuristiques à base de population, plus particulièrement les métaheuristiques d'intelligence en essaims, lors de la résolution de grandes instances de problèmes discrets tels que le MSA. Les méthodes et avenues abordées dans cette thèse tracent ainsi un chemin pour offrir une transposition des techniques utilisées sur davantage de métaheuristiques continues de la littérature.

CHAPITRE 4 – PROPOSITION D'HYBRIDATIONS HOMOGÈNES AVEC LE PSO POUR LA RÉSOLUTION DE PROBLÈMES CONTINUS

Ce chapitre a fait l'objet d'un article de journal :

Deschenes, H., & Gagne, C. (2020). Solving high dimensional multimodal continuous optimisation problems using hybridisation between particle swarm optimisation variants. *International Journal of Metaheuristics*, 7(3), 239-264.
DOI : 10.1504/IJMHEUR.2020.107390

4.1. INTRODUCTION

Ce chapitre traite de la mise en œuvre de stratégies d'hybridation homogènes entre différentes variantes de PSO pour la résolution de fonctions continues. Ces nouvelles hybridations, nommées HCLBPSO-Half, HBPSO+CL et HCoCLPSO, sont conçues en combinant les trois variantes de PSO mentionnées à la Section 3.2.3 : le BPSO (James Kennedy, 2003), le CLPSO (Liang et al., 2006) et le CoLPSO (van den Bergh & Engelbrecht, 2004).

Après avoir détaillé les instances de problèmes continus qui sont utilisées pour les tests, ce chapitre poursuit avec une description de la conception des nouvelles hybridations. Celles-ci sont ensuite testées et analysées sur des fonctions continues provenant de la littérature. La première partie de ces tests concerne la résolution de fonctions classiques (*Sphere*, *Rosenbrock*, *Ackley*, *Griewank*, *Rastrigin* et *Schwefel*) (Dixon & Szego, 1978; Molga & Smutnicki, 2005). Le but visé par cette partie est de valider les bénéfices de l'hybridation lors de la résolution de problèmes avec la comparaison des trois variantes et des trois hybridations conçues. La deuxième partie concerne quant à elle la résolution de fonctions plus complexes, soient 15 fonctions provenant du *CEC'15 benchmark set* (Qu et al., 2016). Dans ce cas, le but est plutôt de valider les performances de la meilleure hybridation provenant des premiers tests en comparaison avec un algorithme provenant de la littérature : dynFWACM (Yu et al., 2015).

4.2. INSTANCES DE PROBLÈMES CONTINUS

La résolution de fonctions continues permet de contribuer à divers problèmes provenant de contextes réels. Ces problèmes peuvent être issus entre autres de la finance informatique, de l'entrainement des réseaux de neurones artificiels, de la prédiction du trafic sur les routes, de la prévision de la température, ou encore de la reconnaissance faciale

(Mavrovouniotis et al., 2017). Pour y parvenir, les méthodes de résolution se comparent par l'utilisation de modèles de fonctions continues provenant de la littérature.

La littérature contient une variété de fonctions continues. Dans le cadre de cette thèse, la première partie des expérimentations utilise six fonctions classiques : *Sphere*, *Rosenbrock*, *Ackley*, *Griewank*, *Rastrigin* et *Schwefel*) (Dixon & Szego, 1978; Molga & Smutnicki, 2005). Alors que *Sphere* et *Rosenbrock* sont de nature unimodale, c'est-à-dire qu'elles ne proposent qu'un seul minimum, les quatre autres fonctions sont plutôt de nature multimodale. Ceci signifie qu'elles proposent de multiples minimums.

Ces fonctions sont sélectionnées afin de tester des méthodes à complexité variables. Elles peuvent être résolues avec une dimension définie à n'importe quel nombre, ce qui en fait des choix intéressants pour la résolution de grandes instances de problèmes. La deuxième partie des expérimentations utilise 15 fonctions provenant du *CEC'15 benchmark set* (Qu et al., 2016). Ces dernières fonctions sont construites en utilisant entre autres des versions altérées des fonctions classiques nommées ci-haut après avoir subi des transformations diverses, des hybridations et des compositions. Il est d'ailleurs plus difficile de résoudre les fonctions provenant de *CEC'15* que de résoudre les fonctions classiques puisqu'elles amènent un niveau de complexité supplémentaire en raison de leur conception élaborée dans le cadre d'un concours.

Les fonctions testées sont divisées selon quatre catégories : unimodales, multimodales, hybrides et par compositions. Le Tableau 4-1 présente les fonctions utilisées dans ce chapitre selon ces catégories. Un numéro unique est également attribué à chacune d'entre elles pour faciliter la comparaison et l'analyse des résultats pour le reste de ce chapitre.

Tableau 4-1 : Fonctions continues utilisées dans ce chapitre selon la catégorie.

Attribut	No.	Nom de la fonction continue
Classique		
Unimodale	A1	*Sphere*
	A2	*Rosenbrock*
Multimodale	A3	*Ackley*
	A4	*Griewank*
	A5	*Rastrigin*
	A6	*Schwefel*
CEC'15 Benchmark set		
Unimodale	B1	*Rotated High Conditioned Elliptic Function*
	B2	*Rotated Cigar Function*
Multimodale	B3	*Shifted and Rotated Ackley's Function*
	B4	*Shifted and Rotated Rastrigin's Function*
	B5	*Shifted and Rotated Schwefel's Function*
Hybride	B6	*Hybrid Function 1 (3 functions)*
	B7	*Hybrid Function 2 (4 functions)*
	B8	*Hybrid Function 3 (5 functions)*
Composition	B9	*Composition Function 1 (3 functions)*
	B10	*Composition Function 2 (3 functions)*
	B11	*Composition Function 3 (5 functions)*
	B12	*Composition Function 4 (5 functions)*
	B13	*Composition Function 5 (5 functions)*
	B14	*Composition Function 6 (7 functions)*
	B15	*Composition Function 7 (10 functions)*

La résolution des fonctions continues est un défi pertinent à relever considérant la difficulté d'équilibrer l'exploration et l'exploitation de l'espace de recherche afin d'obtenir une solution satisfaisante. Pour les résoudre, la prochaine section propose trois nouveaux hybrides construits à partir de variantes de PSO décrites à la Section 3.2.3, soient le BPSO, le CLPSO et le CoLPSO.

4.3. PROPOSITION DE TROIS MODÈLES HYBRIDES DE PSO

Les trois hybridations proposées dans cette section sont conçues dans le but d'améliorer la performance du PSO là où il éprouve plus de difficultés : la résolution de

grandes instances de problèmes et la difficulté à sortir d'un optimum local (Tamayo-Vera et al., 2018; van den Bergh & Engelbrecht, 2004). La première hybridation divise la population en deux sous-essaims distincts. La première moitié de la population se comporte selon le BPSO, et l'autre moitié selon le CLPSO. La deuxième hybridation insère le processus d'apprentissage compréhensif du CLPSO au début d'une itération du BPSO. La troisième hybridation utilise le CoLPSO et ajoute le processus d'apprentissage compréhensif du CLPSO au début de la mise à jour d'une dimension.

Le but principal de ces hybridations est de permettre la poursuite de l'exploration de l'espace de recherche vers la fin de la résolution et d'éviter qu'une particule de rester coincée autour d'un optimum local. Les deux variantes peuvent alors s'entraider en explorant les segments de l'espace de recherche plus difficiles à atteindre pour l'une ou l'autre des variantes lorsqu'elles sont utilisées individuellement. Les prochaines sections décrivent plus en détails les hybrides proposés.

4.3.1. *HCLBPSO-HALF*

La première hybridation élaborée est nommée *HCLBPSO-Half* (*Hybrid Comprehensive Learning Barebones PSO-Half*). Selon la taxonomie de E-G Talbi (2002), elle est classifiée comme étant un algorithme collaboratif de bas niveau (*HCB*). Cette hybridation utilise les bénéfices des deux méthodes (BPSO et CLPSO) simultanément, en séparant la population entière en deux sous-essaims distincts : un premier qui adopte le comportement du BPSO et un deuxième qui adopte le comportement du CLPSO. L'hybridation ainsi produite cherche à utiliser deux comportements différents simultanément et voir à ce que chacune des variantes puisse communiquer à l'autre les solutions trouvées via un vecteur commun, la meilleure solution collective.

La Figure 4-1 suivante illustre cet algorithme hybride. L'assignation entre les deux méthodes sélectionnées est effectuée au commencement de l'algorithme, avant de définir la position initiale des particules dans l'espace de recherche. Chaque particule est mise à jour à chaque itération selon la méthode qui lui est attribuée initialement. Les particules associées au BPSO se mettent à jour selon les encadrés 3-3A et 3-2B, lesquels sont les mêmes que ceux de la Figure 3-3. Les particules associées au CLPSO sont ensuite mises à jour selon les encadrés 3-4A et 3-4B, lesquels sont les mêmes que ceux de la Figure 3-4.

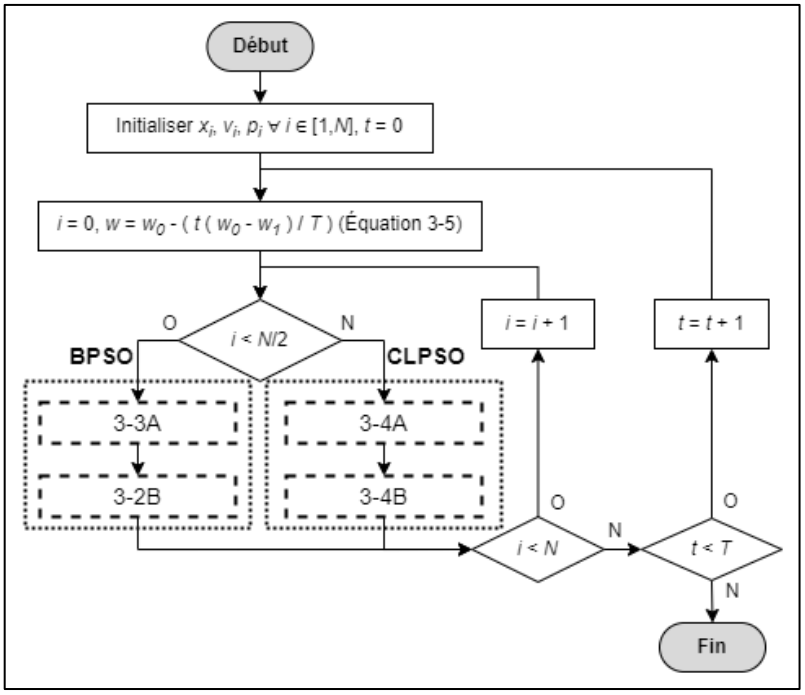

Figure 4-1 : Diagramme organisationnel du *HCLBPSO-Half* avec la division des particules entre le BPSO et le CLPSO.
© Hugo Deschênes

Malgré l'utilisation de deux sous-essaims adoptant une mise à jour différente, l'algorithme ne possède qu'une seule meilleure position collective g. Elle est utilisée à la fois par le CLPSO et par le BPSO. Les deux sous-essaims sont en mesure de s'entraider en profitant des meilleures solutions rencontrées par chacune des variantes. Le BPSO se sert de g pour la génération de la prochaine position d'une particule en calculant les nouvelles valeurs de μ et de σ selon l'Équation 3-8. Considérant que la valeur attribuée à g peut

provenir de n'importe quel sous-essaim, le BPSO met à jour ses particules en profitant des améliorations apportées à g par les particules provenant du CLPSO. De l'autre côté, les particules attribuées au CLPSO peuvent déclencher le processus d'apprentissage compréhensif après M générations stagnantes de solutions. Pendant la sélection par tournoi effectuée durant ce processus, une des deux particules sélectionnées aléatoirement provient du BPSO alors que l'autre provient du CLPSO. Le CLPSO peut alors bénéficier de l'exploration de l'espace de recherche effectuée par les particules du BPSO et des modifications apportées à g effectuées par celles-ci. La meilleure solution globale g est mise à jour en considérant l'ensemble de la population (les deux sous-essaims) après que toutes les particules ont été déplacées et évaluées.

4.3.2. *HBPSO+CL*

La deuxième hybridation est nommée HBPSO+CL (*Hybrid Barebones* PSO avec *Comprehensive Learning*). Selon la taxonomie de E-G Talbi (2002), elle est classifiée comme étant un algorithme collaboratif de haut niveau (HCH). Elle utilise le BPSO comme variante principale et ajoute le processus d'apprentissage compréhensif du CLPSO au commencement de chaque génération. L'hybridation ainsi souhaitée vise à conserver la variabilité de la génération des positions provenant du BPSO et aider les particules à continuer d'explorer l'espace de recherche lorsque la position d'une particule ne contribue pas positivement à la meilleure solution collective pendant plusieurs de générations.

Cette hybridation est illustrée par la Figure 4-2. Dans cet hybride, lors du début de la mise à jour d'une particule, le processus d'apprentissage compréhensif du CLPSO est enclenché dans l'encadré 3-4C. L'hybride poursuit ensuite avec la mise à jour de la position d'une particule selon le BPSO dans l'encadré 3-3A. La particule est ensuite comparée avec sa meilleure solution cognitive et la meilleure solution collective dans l'encadré 3-4B, de manière identique que pour le CLPSO, c'est-à-dire, avec l'ajustement du compteur m_i.

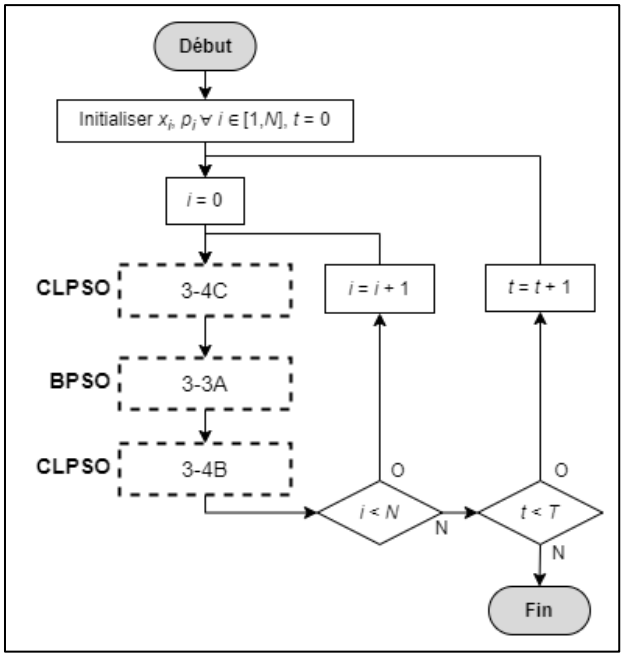

Figure 4-2 : Diagramme organisationnel du HBPSO+CL
© Hugo Deschênes

L'utilisation de l'apprentissage compréhensif à l'intérieur du BPSO lui permet de bénéficier de changements plus radicaux dans la composition des solutions lorsque la valeur de la fonction objectif pour une particule stagne pendant M générations. Même si la génération aléatoire de nombres selon l'(3-8 du BPSO aide à sortir d'un optimum local, le processus d'apprentissage compréhensif aide à intensifier cette habileté et favorise une plus grande exploration de l'espace de recherche. Chaque particule i peut donc avoir une modification apportée à sa meilleure solution cognitive p_i si celle-ci stagne pendant M générations. Cette modification a lieu selon la valeur de la probabilité d'apprentissage Pc_i de la particule provenant de l'Équation 3-10.

4.3.3. *HCOCLPSO*

La troisième hybridation élaborée est nommée HCoCLPSO (*Hybrid Cooperative Comprehensive Learning* PSO). Selon la taxonomie de E-G Talbi (2002), elle est classifiée comme étant un algorithme hybride coopératif de haut niveau (HCH). Elle est inspirée du travail effectué par Maitra et Chatterjee (2008) utilisé pour la résolution du problème de

segmentation d'images à niveaux multiples. Cette méthode ajoute le processus d'apprentissage compréhensif du CLPSO à l'intérieur de la structure du CoLPSO. Ce processus est localisé au début de la mise à jour d'un sous-essaim (d'une dimension). Pour que l'hybride fonctionne correctement, chaque dimension d se voit attribuer un compteur de génération stagnante m_d. Si les particules d'un sous-essaim n'améliorent pas la meilleure solution collective g pendant M générations, le compteur est incrémenté de 1. Autrement, le compteur est réinitialisé à 0.

La Figure 4-3 illustre cette hybridation, où l'encadré 4-3D met en évidence la partie de l'hybride contenant la mise à jour d'un sous-essaim. Cette mise à jour débute avec l'encadré 4-3C, lequel contient le déclenchement du nouveau processus d'apprentissage compréhensif dans le ② si la valeur du compteur de générations stagnantes m_d dépasse la valeur critique M. Une fois ce processus complété, l'hybride poursuit avec la mise à jour de toutes les particules du sous-essaim. Pendant cette mise à jour, chaque particule se déplace et copie ensuite la valeur de sa nouvelle position x_{id}^{t+1} au sein du vecteur contextuel cv. Ce vecteur est ensuite évalué et comparé avec la performance de la meilleure solution cognitive p_{id} de la particule, de même qu'avec la meilleure solution collective de l'essaim g. Une fois que l'évaluation de toutes les particules du sous-essaim est complétée, le compteur m_d est modifié selon l'amélioration ou non de la meilleure solution collective.

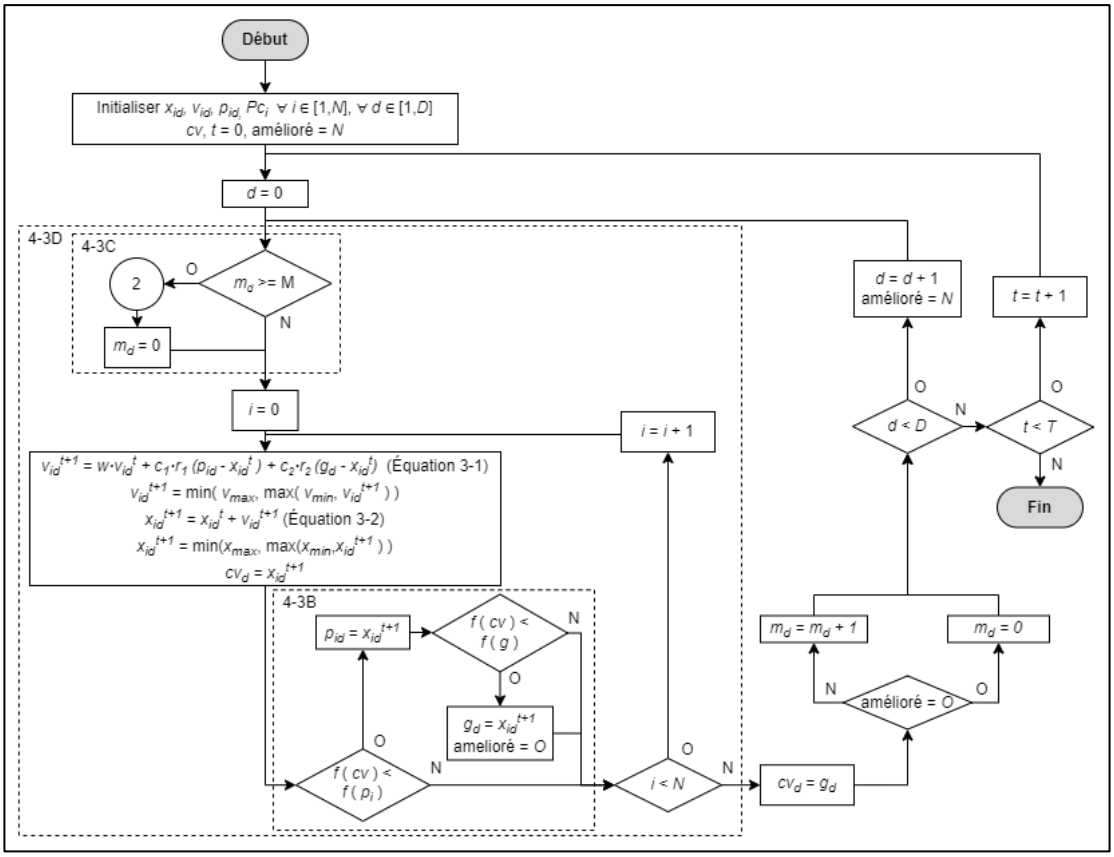

Figure 4-3 : Diagramme organisationnel du HCoCLPSO
© Hugo Deschênes

L'attrait principal de cette hybridation se situe au niveau du processus d'apprentissage compréhensif provenant du CLPSO. Il ne s'agit plus d'une sélection par tournoi des particules comme à la Figure 3-5. L'hybridation effectue plutôt une recherche de la particule ayant la pire valeur de la meilleure solution cognitive selon l'évaluation par la fonction objectif. Cette valeur est remplacée par celle correspondant à la meilleure solution cognitive d'une autre particule provenant du même sous-essaim. Ce processus peut être activé pour chaque dimension du problème à résoudre lorsque ladite dimension n'améliore pas la meilleure solution collective de l'essaim g pendant M générations. Également, la probabilité d'apprentissage Pc_i est ignorée pour cette hybridation afin d'apporter des changements plus fréquents aux particules. Le nouveau processus d'apprentissage compréhensif est illustré à la Figure 4-4.

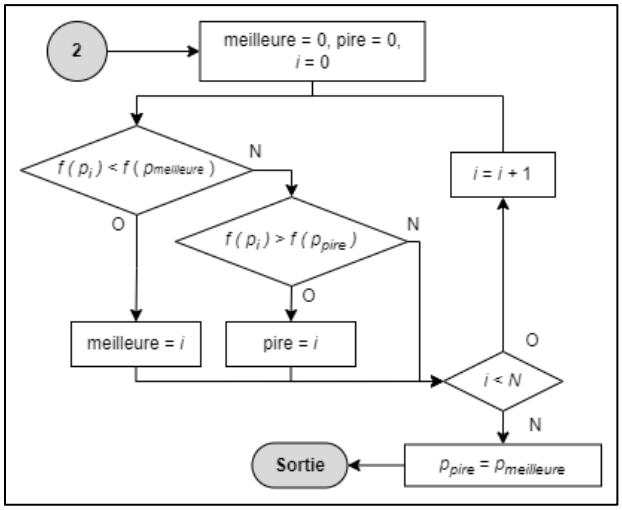

Figure 4-4 : Nouveau processus d'apprentissage compréhensif pour le HCoCLPSO
© Hugo Deschênes

Le principal avantage du CoLPSO est que cette variante permet la résolution d'un problème en travaillant avec chaque dimension individuellement. Ajouter un processus d'apprentissage compréhensif au début de la mise à jour d'un sous-essaim permet au CoLPSO de favoriser l'exploration de l'espace de recherche lorsque les changements apportés à une dimension n'améliorent pas la qualité des solutions. Chaque dimension est alors travaillée de manière indépendante, ce qui fait que les mauvaises performances d'une des dimensions ne viennent pas impacter négativement les performances d'une autre dimension.

4.4. EXPÉRIMENTATIONS NUMÉRIQUES

Pour valider l'efficacité des hybridations élaborées à la section précédente, les six fonctions continues classiques (A1 à A6) et les quinze fonctions continues avancées (B1 à B15) décrites à la Section 4.2 sont utilisées. Chacune des fonctions continues est configurée afin d'effectuer une résolution où les variables de décision sont incluses dans l'intervalle $[-100, 100]$. Les expérimentations sont effectuées pour une dimension définie à 50, 100 et 200 pour chacune des fonctions classiques, et pour une dimension définie à 10, 30, 50 et

100 pour les fonctions provenant de *CEC'15*. Les tests sont exécutés sur un ordinateur comportant le système d'exploitation Windows 10 Professionnel, avec un processeur Intel® Core™ i5-2400 incluant un CPU à 3.10GHz et 8,00 Go de mémoire vive.

Les variantes et les hybridations utilisent un nombre total de particules (N) défini à 40, à l'exception du CoLPSO et du HCoCLPSO. Cette valeur est sélectionnée considérant le travail effectué par Pluhacek et al. (2016), où ils suggèrent que les variantes de PSO performent bien avec l'utilisation de 40 particules. Ce nombre est plus grand pour le CoLPSO et le HCoCLPSO étant donné que leur structure implique l'utilisation d'un nombre de sous-essaims égal à la dimension du problème utilisé, et par conséquent, le nombre d'évaluations par itération est plus grand. Après avoir effectué des tests empiriques avec ces 2 algorithmes, nous avons déterminé que l'utilisation d'une population de 20 particules par sous-essaim apporte des résultats légèrement meilleurs que si 40 particules étaient utilisées. Les paramètres de confiance c_1 et c_2 sont chacun définis à 1.49445, un nombre démontré efficace pour le PSO (R. C. Eberhart & Shi, 2000). Le paramètre d'inertie w varie selon la diminution dynamique proposée par Ratnaweera et al. (2004), laquelle correspond à l'utilisation de l'Équation 3-5. Le paramètre w_0 est donc défini à 0.9, et le paramètre w_1 à 0.4. Ceci permet à l'algorithme d'avoir une bonne transition entre l'exploration et l'exploitation de l'espace de recherche à mesure que l'algorithme approche de la dernière génération. Ces paramètres sont définis selon une étude effectuée par Schutte et Groenwold (2005) mentionnant que la diminution dynamique de l'inertie est moins sensible à la variation des paramètres du PSO. Ils ont également prouvé qu'il fonctionne mieux avec l'utilisation de problèmes comportant une grande dimension.

La comparaison entre les hybridations est réalisée d'une manière similaire à ce qui est proposé par Yu et al. (2015). Ces auteurs définissent le nombre total dévaluations par la fonction objectif à 10 000 fois multiplié par la dimension du problème. Les tests ont été exécutés 50 fois pour chaque méthode et pour chaque fonction continue, tout en retenant la

valeur de la meilleure solution globale g pour toutes les exécutions. La moyenne et la médiane des exécutions sont également calculées, permettant ainsi de comparer les résultats des hybridations avec les variantes utilisées pour les construire. La meilleure méthode parmi les six est ensuite comparée avec l'algorithme proposé par Yu et al. (2015). Afin de comparer les résultats pour la résolution des fonctions provenant de *CEC'15* avec ceux obtenus par Yu et al. dans leur article, les auteurs proposent une modification aux solutions. Plus précisément, la valeur 100 est ajoutée aux résultats de la fonction 1, 200 à la fonction 2, 300 à la fonction 3, et ainsi de suite pour les 15 fonctions continues. Ces valeurs sont soustraites des tableaux présentés à la section suivante afin que la meilleure valeur demeure 0 au lieu des incrémentations suggérées. Afin de présenter l'analyse des résultats, les sous-sections suivantes abordent en premier lieu l'analyse selon des fonctions classiques, et en deuxième lieu l'analyse des résultats des fonctions *CEC'15*, avant de poursuivre en dernier lieu avec les conclusions à tirer de ces expérimentations.

4.4.1. RÉSULTATS ET ANALYSE DE LA RÉSOLUTION DES FONCTIONS CONTINUES CLASSIQUES

La première partie des expérimentations concerne la comparaison des résultats pour la résolution des fonctions continues classiques, soient les deux fonctions unimodales *Sphere* (A1), *Rosenbrock* (A2), de même que les quatre fonctions multimodales *Ackley* (A3), *Griewwank* (A4), *Rastrigin* (A5) et *Schwefel* (A6). Pour ce faire, quatre tableaux sont présentés dans cette section illustrant une comparaison des résultats entre les variantes ayant servies aux hybridations et les hybrides proposés à la Section 4.3. Pour faciliter la comparaison, toutes les valeurs obtenues inférieures à $1 \cdot 10^{-15}$ sont arrondies à 0 et sont considérées équivalentes entre elles. Une analyse des résultats est ensuite décrite à la fin de cette section.

Le Tableau 4-2 présente la performance des variantes du PSO (BPSO, CLPSO et CoLPSO) et le Tableau 4-3 présente la performance des trois hybridations proposées (HBPSO+CL, HCLBPSO-Half et HCoCLPSO). Ces tableaux affichent la meilleure et la pire solution obtenues pour chaque problème pour chacune des méthodes, de même que la médiane, la moyenne et l'écart-type pour l'ensemble des 50 exécutions. Les valeurs en gras italique représentent la meilleure valeur obtenue par donnée statistique.

Le Tableau 4-4 qui suit reprend les résultats du Tableau 4-2 et du Tableau 4-3, en mettant l'emphase sur la comparaison entre les variantes de PSO et les hybridations proposées selon les médianes et les moyennes. Les nombres en gras italiques représentent les meilleures valeurs pour chacun des problèmes sur les six méthodes continues classiques, selon la moyenne et la médiane des valeurs obtenues. Les nombres soulignés seulement représentent la deuxième meilleure valeur pour les mêmes statistiques. Le Tableau 4-4 aide à déterminer quel algorithme performe le mieux sur les fonctions continues classiques.

Tableau 4-2 : Résultats des variantes BPSO, CLPSO et CoLPSO lors de la résolution des fonctions continues classiques pour les dimensions 50, 100 et 200.

		50D														
		BPSO					CLPSO					CoLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	1,45E-01	7,27E-01	3,83E-01	3,95E-01	1,24E-01	6,31E-13	5,06E-10	9,34E-12	2,54E-11	7,23E-11
	A2	***4,10E-01***	1,01E+04	***3,52E+01***	3,21E+02	1,44E+03	4,05E+03	2,22E+04	1,03E+04	1,07E+04	3,97E+03	2,43E+00	***2,09E+02***	9,71E+01	***9,79E+01***	***5,26E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,01E+01	2,00E+01	2,00E+01	1,20E-02	***1,09E-06***	***8,17E-05***	***5,19E-06***	***1,06E-05***	1,44E-05
	A4	***0,00E+00***	2,67E+00	3,70E-03	1,62E-01	6,11E-01	3,36E-03	***3,63E-02***	8,40E-03	1,01E-02	***5,76E-03***	4,58E-13	9,23E-02	***5,90E-11***	***8,91E-03***	1,84E-02
	A5	8,36E+01	3,01E+02	1,53E+02	1,68E+02	4,99E+01	1,50E+01	4,65E+01	2,56E+01	2,53E+01	6,52E+00	***9,14E-09***	***5,45E-06***	***9,53E-08***	***3,88E-07***	***8,70E-07***
	A6	1,80E+04	1,82E+04	1,81E+04	1,81E+04	4,71E+01	***1,78E+04***	1,80E+04	***1,78E+04***	***1,78E+04***	4,17E+01	***1,78E+04***	***1,78E+04***	***1,78E+04***	***1,78E+04***	***1,10E-11***
		100D														
		BPSO					CLPSO					CoLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	4,82E-02	1,94E-01	1,07E-01	1,11E-01	3,05E-02	7,41E-12	6,02E-10	5,33E-11	9,23E-11	1,20E-10
	A2	***1,46E-01***	1,30E+03	***1,51E+02***	2,88E+02	3,61E+02	1,65E+03	1,00E+04	3,52E+03	3,97E+03	1,83E+03	7,18E+01	***3,42E+02***	1,79E+02	***1,86E+02***	***6,62E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,01E+01	2,01E+01	2,01E+01	2,37E-02	***6,07E-06***	***1,07E-03***	***2,85E-05***	***7,61E-05***	1,63E-04
	A4	***0,00E+00***	7,74E+00	2,74E+00	2,67E+00	2,20E+00	6,28E-04	***8,29E-03***	1,18E-03	***1,38E-03***	***1,06E-03***	3,71E-12	5,62E-02	***2,12E-09***	6,29E-03	1,09E-02
	A5	2,53E+02	7,92E+02	4,34E+02	4,52E+02	1,22E+02	7,50E+01	1,60E+02	1,03E+02	1,05E+02	1,87E+01	***6,26E-07***	***3,85E-03***	***6,57E-06***	***1,71E-04***	***6,43E-04***
	A6	3,60E+04	3,64E+04	3,61E+04	3,61E+04	6,56E+01	3,56E+04	3,60E+04	3,58E+04	3,58E+04	8,31E+01	***3,55E+04***	***3,55E+04***	***3,55E+04***	***3,55E+04***	***2,20E-11***
		200D														
		BPSO					CLPSO					CoLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***2,00E-32***	***8,88E-09***	***0,00E+00***	***1,78E-10***	***1,26E-09***	1,72E-02	1,40E-01	4,02E-02	4,61E-02	2,35E-02	8,80E-11	2,03E-08	5,70E-10	1,39E-09	3,00E-09
	A2	***4,10E-01***	1,05E+04	***3,53E+02***	1,18E+03	2,77E+03	1,98E+03	1,16E+04	3,01E+03	3,70E+03	1,97E+03	2,29E+02	***5,32E+02***	3,70E+02	***3,76E+02***	***7,48E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,02E+01	2,01E+01	2,01E+01	3,77E-02	***3,19E-05***	***2,58E-02***	***1,90E-04***	***1,58E-03***	4,81E-03
	A4	***3,66E-15***	7,55E+00	3,38E+00	3,24E+00	1,93E+00	7,69E-05	***8,17E-03***	2,12E-04	***3,81E-04***	***1,13E-03***	1,90E-11	3,19E-02	***2,88E-10***	3,54E-03	7,29E-03
	A5	4,63E+02	1,49E+03	7,50E+02	8,09E+02	2,59E+02	3,25E+02	6,31E+02	4,56E+02	4,61E+02	6,33E+01	***4,56E-04***	***5,74E-01***	***7,70E-03***	***4,85E-02***	***1,11E-01***
	A6	7,22E+04	7,25E+04	7,23E+04	7,23E+04	9,31E+01	7,20E+04	7,32E+04	7,24E+04	7,25E+04	2,33E+02	***7,11E+04***	***7,11E+04***	***7,11E+04***	***7,11E+04***	***4,41E-11***

Tableau 4-3 : Résultats des hybrides HCLBPSO-Half, HBPSO+CL et HCoCLPSO lors de la résolution des fonctions continues classiques pour les dimensions 50, 100 et 200.

		50D														
		HBPSO-CL					HCLBPSO-Half					HCOCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	6,50E-09	1,08E-07	2,92E-08	3,48E-08	2,13E-08
	A2	1,01E-01	2,57E+02	***2,14E+01***	***3,88E+01***	5,58E+01	***5,56E-02***	1,14E+03	7,07E+01	1,45E+02	2,82E+02	1,17E+00	***2,28E+02***	9,67E+01	9,84E+01	***5,18E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	***2,87E-05***	***1,94E-04***	***7,61E-05***	***8,29E-05***	3,38E-05
	A4	***0,00E+00***	2,88E+00	***0,00E+00***	1,57E-01	5,99E-01	***0,00E+00***	1,34E-01	***0,00E+00***	1,45E-02	2,99E-02	1,64E-09	***2,70E-02***	2,41E-08	***5,96E-03***	***8,32E-03***
	A5	8,95E+01	2,73E+02	1,68E+02	1,73E+02	4,12E+01	6,57E+01	1,92E+02	1,22E+02	1,25E+02	3,14E+01	***3,07E-06***	***6,39E-05***	***1,33E-05***	***1,74E-05***	***1,32E-05***
	A6	1,80E+04	1,82E+04	1,81E+04	1,81E+04	4,27E+01	1,79E+04	1,83E+04	1,81E+04	1,81E+04	6,49E+01	***1,78E+04***	***1,78E+04***	***1,78E+04***	***1,78E+04***	***1,10E-11***
		100D														
		HBPSO-CL					HCLBPSO-Half					HCOCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	2,80E-08	2,81E-07	7,86E-08	9,22E-08	5,86E-08
	A2	***5,19E+00***	1,20E+03	***1,31E+02***	2,50E+02	3,25E+02	1,31E+01	1,14E+03	1,53E+02	2,04E+02	2,16E+02	5,60E+00	***3,76E+02***	1,97E+02	***2,00E+02***	***6,21E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	***4,91E-05***	***2,38E-04***	***9,88E-05***	***1,05E-04***	3,51E-05
	A4	***0,00E+00***	7,46E+00	2,48E+00	2,75E+00	1,66E+00	***0,00E+00***	2,03E-01	***0,00E+00***	1,53E-02	3,65E-02	1,21E-09	***2,46E-02***	2,21E-08	***4,83E-03***	***7,39E-03***
	A5	2,54E+02	7,12E+02	4,45E+02	4,57E+02	1,06E+02	2,67E+02	6,30E+02	4,20E+02	4,41E+02	8,89E+01	***1,49E-05***	***1,51E-04***	***4,64E-05***	***5,35E-05***	***2,93E-05***
	A6	3,60E+04	3,63E+04	3,62E+04	3,62E+04	7,36E+01	3,60E+04	3,64E+04	3,62E+04	3,62E+04	8,58E+01	***3,55E+04***	***3,55E+04***	***3,55E+04***	***3,55E+04***	***2,20E-11***
		200D														
		HBPSO-CL					HCLBPSO-Half					HCOCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***0,00E+00***	2,91E-06	***0,00E+00***	5,83E-08	4,11E-07	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	1,08E-07	7,22E-07	1,78E-07	2,07E-07	1,03E-07
	A2	2,21E+02	1,05E+04	4,04E+02	1,71E+03	3,26E+03	***1,49E+02***	1,04E+04	3,88E+02	8,89E+02	2,00E+03	1,84E+02	***5,67E+02***	***3,76E+02***	***3,81E+02***	***7,40E+01***
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***0,00E+00***	***6,40E-05***	***2,19E-04***	***1,04E-04***	***1,09E-04***	2,65E-05
	A4	2,03E-01	6,96E+00	3,40E+00	3,46E+00	1,38E+00	***1,11E-15***	3,36E+00	***1,13E-14***	1,92E-01	6,62E-01	3,34E-09	***1,97E-02***	1,01E-08	***1,68E-03***	***4,18E-03***
	A5	4,98E+02	1,10E+03	7,45E+02	7,67E+02	1,55E+02	9,09E+02	2,09E+03	1,34E+03	1,35E+03	2,54E+02	***8,67E-05***	***5,57E-04***	***2,13E-04***	***2,11E-04***	***8,46E-05***
	A6	7,22E+04	7,26E+04	7,23E+04	7,23E+04	9,99E+01	7,21E+04	7,26E+04	7,24E+04	7,24E+04	1,12E+02	***7,11E+04***	***7,11E+04***	***7,11E+04***	***7,11E+04***	***4,41E-11***

Tableau 4-4 : Résultats combinés variantes et des hybrides lors de la résolution des fonctions continues classiques pour les dimensions 50, 100 et 200.

		50D											
		BPSO		CLPSO		CoLPSO		HBPSO+CL		HCLBPSO-Half		HCoCLPSO	
		Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.
Unimodale	A1	***0,00E+00***	***0,00E+00***	3,83E-01	3,95E-01	9,34E-12	2,54E-11	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	2,92E-08	3,48E-08
	A2	3,52E+01	3,21E+02	1,03E+04	1,07E+04	9,71E+01	9,79E+01	***2,14E+01***	***3,88E+01***	7,07E+01	1,45E+02	9,67E+01	9,84E+01
Multimodale	A3	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***5,19E-06***	***1,06E-05***	2,00E+01	2,00E+01	2,00E+01	2,00E+01	7,61E-05	8,29E-05
	A4	3,70E-03	1,62E-01	8,40E-03	1,01E-02	5,90E-11	8,91E-03	***0,00E+00***	1,57E-01	***0,00E+00***	1,45E-02	2,41E-08	***5,96E-03***
	A5	1,53E+02	1,68E+02	2,56E+01	2,53E+01	***9,53E-08***	***3,88E-07***	1,68E+02	1,73E+02	1,22E+02	1,25E+02	1,33E-05	1,74E-05
	A6	1,81E+04	1,81E+04	***1,78E+04***	***1,78E+04***	***1,78E+04***	***1,78E+04***	1,81E+04	1,81E+04	1,81E+04	1,81E+04	***1,78E+04***	***1,78E+04***
		100D											
		BPSO		CLPSO		CoLPSO		HBPSO+CL		HCLBPSO-Half		HCoCLPSO	
		Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.
Unimodale	A1	***0,00E+00***	***0,00E+00***	1,07E-01	1,11E-01	5,33E-11	9,23E-11	***0,00E+00***	***0,00E+00***	***0,00E+00***	***0,00E+00***	7,86E-08	9,22E-08
	A2	1,51E+02	2,88E+02	3,52E+03	3,97E+03	1,79E+02	***1,86E+02***	***1,31E+02***	2,50E+02	1,53E+02	2,04E+02	1,97E+02	2,00E+02
Multimodale	A3	2,00E+01	2,00E+01	2,01E+01	2,01E+01	***2,85E-05***	***7,61E-05***	2,00E+01	2,00E+01	2,00E+01	2,00E+01	9,88E-05	1,05E-04
	A4	2,74E+00	2,67E+00	1,18E-03	***1,38E-03***	2,12E-09	6,29E-03	2,48E+00	2,75E+00	***0,00E+00***	1,53E-02	2,21E-08	4,83E-03
	A5	4,34E+02	4,52E+02	1,03E+02	1,05E+02	***6,57E-06***	1,71E-04	4,45E+02	4,57E+02	4,20E+02	4,41E+02	4,64E-05	***5,35E-05***
	A6	3,61E+04	3,61E+04	3,58E+04	3,58E+04	***3,55E+04***	***3,55E+04***	3,62E+04	3,62E+04	3,62E+04	3,62E+04	***3,55E+04***	***3,55E+04***
		200D											
		BPSO		CLPSO		CoLPSO		HBPSO+CL		HCLBPSO-Half		HCoCLPSO	
		Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.	Méd.	Moy.
Unimodale	A1	***0,00E+00***	1,78E-10	4,02E-02	4,61E-02	5,70E-10	1,39E-09	***0,00E+00***	5,83E-08	***0,00E+00***	***0,00E+00***	1,78E-07	2,07E-07
	A2	***3,53E+02***	1,18E+03	3,01E+03	3,70E+03	3,70E+02	***3,76E+02***	4,04E+02	1,71E+03	3,88E+02	8,89E+02	3,76E+02	3,81E+02
Multimodale	A3	2,00E+01	2,00E+01	2,01E+01	2,01E+01	1,90E-04	1,58E-03	2,00E+01	2,00E+01	2,00E+01	2,00E+01	***1,04E-04***	***1,09E-04***
	A4	3,38E+00	3,24E+00	2,12E-04	***3,81E-04***	2,88E-10	3,54E-03	3,40E+00	3,46E+00	***1,13E-14***	1,92E-01	1,01E-08	1,68E-03
	A5	7,50E+02	8,09E+02	4,56E+02	4,61E+02	7,70E-03	4,85E-02	7,45E+02	7,67E+02	1,34E+03	1,35E+03	***2,13E-04***	***2,11E-04***
	A6	7,23E+04	7,23E+04	7,24E+04	7,25E+04	***7,11E+04***	***7,11E+04***	7,23E+04	7,23E+04	7,24E+04	7,24E+04	***7,11E+04***	***7,11E+04***

Le Tableau 4-4 démontre que la variante CoLPSO est la méthode qui performe mieux que les autres pour une dimension définie à 50 et à 100. Cette variante a obtenu à 8 reprises les meilleures valeurs de médianes et à huit reprises les meilleures valeurs de moyennes pour ces dimensions, sur l'ensemble des 18 tests. L'hybridation HCoCLPSO est la deuxième méthode la plus performante avec des résultats très près du CoLPSO lorsqu'elle est comparée aux cinq autres algorithmes. Elle a quant à elle obtenue à cinq reprises les meilleures valeurs de médianes et à sept reprises les meilleures valeurs de moyennes. Cependant, l'hybride HCoCLPSO est la méthode démontrant les meilleures performances lorsqu'une dimension de 200 est utilisée étant donné le plus grand nombre de meilleures médianes et de meilleures moyennes en comparaison avec les autres algorithmes. Les deux autres hybrides, HCLBPSO-Half et HBPSO+CL, offrent tout de même des performances intéressantes pour les problèmes de nature unimodale. Elles sont tout de même moins performantes que le troisième hybride HCoCLPSO.

L'hybride HCoCLPSO obtient de meilleurs résultats plus la dimension d'un problème augmente. Lorsque celle-ci est définie à 100, l'hybridation débute à démontrer des résultats prometteurs en obtenant à six reprises la deuxième meilleure valeur maximale, et à trois reprises la meilleure valeur de médiane et de moyenne. Sur les problèmes comportant une dimension définie à 200, elle obtient à deux reprises la deuxième meilleure valeur maximale et à six reprises la meilleure valeur de médiane et de moyenne. Également, toutes les meilleures valeurs maximales obtenues concernent les fonctions multimodales, ce qui correspond à une catégorie de problèmes plus difficile à résoudre. L'hybride HCoCLPSO est donc meilleur que les autres algorithmes expérimentés lorsque les dimensions d'un problème augmentent, contribuant ainsi efficacement à la résolution de problèmes continus de grande taille.

Les résultats obtenus avec la variante CoLPSO et l'hybride HCoCLPSO démontrent l'efficacité de la résolution de chaque dimension de manière indépendante lors de la

résolution de fonctions continues multimodales. La dominance du HCoCLPSO sur le CoLPSO sur les problèmes avec une dimension définie à 200 démontre également que l'utilisation du processus d'apprentissage compréhensif permettant de sortir d'un optimum local est une bonne stratégie pour continuer à améliorer les résultats lors de la résolution de grandes instances de problèmes.

Afin de mieux démontrer la contribution de chaque variante, le Tableau 4-5 présente plus de détails sur les résultats des tests exécutés avec une dimension définie à 200 pour l'hybride HCoCLPSO et les deux variantes à partir desquelles cette hybridation est élaborée (le CLPSO et le CoLPSO). Dans ce tableau, les nombres en gras italique représentent les meilleures valeurs pour chaque donnée statistique, pour chaque instance de problème, à l'exception de la statistique sur la pire donnée obtenue. Dans ce cas, les données en gras italique représentent les pires valeurs.

Les valeurs surlignées au Tableau 4-5 démontrent que l'hybride HCoCLPSO performe mieux que ses deux variantes à partir desquelles elle est conçue pour la résolution des fonctions multimodales (A3 à A6). En effet, elle possède à 3 reprises le plus grand nombre de meilleures médianes et à 3 reprises le plus grand nombre de meilleures moyennes. Elle ne possède pas le plus grand nombre de meilleures valeurs maximales avec un score de 2, mais elle est tout de même la méthode la plus constante considérant qu'elle obtient à 3 reprises le plus grand nombre de meilleurs écarts-types. Elle est également la méthode ayant obtenu le plus faible nombre de pires valeurs obtenues avec un score de 0. À la lumière de ces résultats, l'hybridation HCoCLPSO est la méthode démontrant le plus grand potentiel sur la résolution de fonctions classiques continues de grande taille selon le contexte des expérimentations. Plus de tests se poursuivent à la section suivante avec la résolution de fonctions plus complexes provenant du *CEC'15 benchmark set*.

Tableau 4-5 : Résultats détaillés de la résolution des fonctions classiques continues avec une dimension définie à 200 pour l'hybride HCoCLPSO et ses variantes qui le composent

		200D				
		CLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	1,72E-02	1,40E-01	4,02E-02	4,61E-02	2,35E-02
	A2	1,98E+03	1,16E+04	3,01E+03	3,70E+03	1,97E+03
Multimodale	A3	2,00E+01	2,02E+01	2,01E+01	2,01E+01	3,77E-02
	A4	7,69E-05	***8,17E-03***	2,12E-04	***3,81E-04***	***1,13E-03***
	A5	3,25E+02	6,31E+02	4,56E+02	4,61E+02	6,33E+01
	A6	7,20E+04	7,32E+04	7,24E+04	7,25E+04	2,33E+02
		CoLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	***8,80E-11***	***2,03E-08***	***5,70E-10***	***1,39E-09***	***3,00E-09***
	A2	2,29E+02	***5,32E+02***	***3,70E+02***	***3,76E+02***	7,48E+01
Multimodale	A3	***3,19E-05***	2,58E-02	1,90E-04	1,58E-03	4,81E-03
	A4	***1,90E-11***	3,19E-02	***2,88E-10***	3,54E-03	7,29E-03
	A5	4,56E-04	5,74E-01	7,70E-03	4,85E-02	1,11E-01
	A6	***7,11E+04***	***7,11E+04***	***7,11E+04***	***7,11E+04***	***4,41E-11***
		HCoCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	A1	1,08E-07	7,22E-07	1,78E-07	2,07E-07	1,03E-07
	A2	***1,84E+02***	5,67E+02	3,76E+02	3,81E+02	***7,40E+01***
Multimodale	A3	6,40E-05	***2,19E-04***	***1,04E-04***	***1,09E-04***	***2,65E-05***
	A4	3,34E-09	1,97E-02	1,01E-08	1,68E-03	4,18E-03
	A5	***8,67E-05***	***5,57E-04***	***2,13E-04***	***2,11E-04***	***8,46E-05***
	A6	***7,11E+04***	***7,11E+04***	***7,11E+04***	***7,11E+04***	***4,41E-11***

4.4.2. RÉSULTATS ET ANALYSE DE LA RÉSOLUTION DES FONCTIONS CONTINUES DE *CEC'15*

La deuxième partie des expérimentations concerne la comparaison des résultats pour la résolution des fonctions continues provenant de *CEC'15*, soient deux fonctions unimodales (B1 et B2), trois fonctions multimodales (B3 à B5), trois fonctions hybrides (B6 à B8) et sept fonctions par compositions (B9 à B15). Pour ce faire, cinq tableaux sont présentés dans cette section illustrant une comparaison des résultats entre la meilleure hybridation de la section précédente (HCoCLPSO) et dynFWACM, lequel est l'algorithme proposé par Yu et al. (2015). Une analyse des résultats est ensuite décrite à la fin de cette section.

Chaque valeur de dimension (taille du problème) testée est présentée séparément au Tableau 4-6 (10D), Tableau 4-7 (30D), Tableau 4-8 (50D) et au Tableau 4-9 (100D). Ces valeurs sont les mêmes que celles utilisées par Yu et al. dans leur article. La meilleure solution, la pire solution, la médiane, la moyenne et l'écart-type sont présentés pour chacune des quinze fonctions pour les 50 exécutions. Les valeurs en gras italiques dans ces tableaux représentent les meilleurs résultats obtenus pour chaque donnée statistique. Les valeurs en gras italiques pour les pires solutions représentent quant à elles les pires résultats obtenus. La dernière ligne de chaque tableau, nommée *Total*, montre le nombre de fois où une méthode a obtenu la valeur mise en évidence pour chaque donnée statistique.

Tableau 4-6 : Comparaison entre dynFWACM (Yu et al., 2015) et HCoCLPSO pour une dimension définie à 10 sur les problèmes *CEC'15.*

		10D									
		dynFWACM					HCoCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	**B1**	7,05E+03	***4,90E+05***	9,23E+04	1,11E+05	***9,88E+04***	***1,50E+03***	1,50E+03	***1,50E+03***	***1,50E+03***	5,77E+05
	B2	1,75E+01	3,30E+04	***6,26E+03***	***8,86E+03***	***8,90E+03***	***7,45E+00***	***3,36E+04***	6,33E+03	1,11E+04	1,10E+04
Multimodale	**B3**	***2,00E+01***	2,00E+01	***2,00E+01***	2,00E+01	***2,33E-04***	2,01E+00	2,00E+01	***2,00E+01***	***1,96E+01***	2,54E+00
	B4	6,96E+00	3,48E+01	***1,59E+01***	***1,67E+01***	***6,65E+00***	***5,97E+00***	***4,48E+01***	1,89E+01	2,07E+01	8,94E+00
	B5	1,30E+02	1,09E+03	***4,78E+02***	***5,18E+02***	2,40E+02	***1,19E+02***	***1,17E+03***	5,53E+02	5,92E+02	***2,13E+02***
Hybride	**B6**	***3,59E+01***	7,31E+03	***8,30E+02***	***1,74E+03***	***1,97E+03***	3,06E+03	***5,57E+04***	8,54E+03	1,33E+04	1,20E+04
	B7	9,03E-01	***2,12E+00***	1,45E+00	1,45E+00	***2,31E-01***	***2,57E-01***	1,71E+00	***1,20E+00***	***1,22E+00***	3,29E-01
	B8	***1,23E+01***	8,03E+03	***1,24E+03***	***1,96E+03***	***2,13E+03***	1,12E+03	***1,15E+06***	4,15E+04	1,57E+05	2,57E+05
Composition	**B9**	***1,00E+02***	1,02E+02	***1,00E+02***	***1,00E+02***	***4,59E-01***	***1,00E+02***	***2,29E+02***	***1,00E+02***	1,22E+02	4,69E+01
	B10	***1,54E+02***	1,22E+03	***4,75E+02***	***5,47E+02***	***2,37E+02***	2,99E+02	***1,26E+05***	3,92E+03	1,10E+04	2,18E+04
	B11	***1,85E+00***	3,03E+02	***3,01E+02***	***1,85E+02***	1,47E+02	1,84E+02	***6,06E+02***	3,03E+02	3,42E+02	***9,32E+01***
	B12	1,08E+02	1,17E+02	1,13E+02	***1,13E+02***	**1,52E+00**	***1,02E+02***	***2,31E+02***	***1,04E+02***	1,15E+02	3,52E+01
	B13	***9,60E-02***	1,96E-01	***1,20E-01***	***1,21E-01***	***1,66E-02***	2,47E+01	***1,25E+03***	3,48E+01	5,82E+01	1,72E+02
	B14	***1,00E+02***	8,74E+03	***6,83E+03***	6,30E+03	***2,12E+03***	***1,00E+02***	***1,14E+04***	7,00E+03	***5,81E+03***	2,92E+03
	B15	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	1,44E-13	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	***0,00E+00***
TOTAL		9	2	12	11	12	9	11	6	5	3

Tableau 4-7 : Comparaison entre dynFWACM (Yu et al., 2015) et HCoCLPSO pour une dimension définie à 30 sur les problèmes *CEC'15.*

		30D									
		dynFWACM					HCoCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	B1	***2,55E+05***	1,34E+06	***5,57E+05***	***6,17E+05***	***3,49E+05***	6,60E+05	***6,37E+06***	3,02E+06	3,18E+06	1,43E+06
	B2	***2,93E+01***	1,46E+04	***2,15E+03***	***3,31E+03***	***3,59E+03***	3,64E+01	***1,50E+04***	3,44E+03	4,71E+03	4,19E+03
Multimodale	B3	***2,00E+01***	2,00E+01	***2,00E+01***	***2,00E+01***	***5,05E-06***	***2,00E+01***	***2,01E+01***	***2,00E+01***	***2,00E+01***	8,91E-03
	B4	7,26E+01	***2,55E+02***	1,21E+02	1,30E+02	3,80E+01	***4,68E+01***	1,89E+02	***1,00E+02***	***1,00E+02***	***2,98E+01***
	B5	1,96E+03	***4,88E+03***	3,32E+03	3,38E+03	6,98E+02	***1,55E+03***	4,21E+03	***2,82E+03***	***2,75E+03***	***5,25E+02***
Hybride	B6	***2,91E+03***	8,35E+04	***2,06E+04***	***2,69E+04***	***1,90E+04***	1,03E+05	***3,23E+06***	6,55E+05	8,15E+05	7,30E+05
	B7	8,58E+00	2,00E+01	1,49E+01	1,46E+01	***2,57E+00***	***4,36E+00***	***8,04E+01***	***8,98E+00***	***1,24E+01***	1,44E+01
	B8	***3,86E+03***	5,00E+04	***2,25E+04***	***2,40E+04***	***1,32E+04***	4,33E+04	***1,53E+06***	3,99E+05	5,10E+05	3,79E+05
Composition	B9	1,06E+02	1,10E+02	1,08E+02	***1,08E+02***	***9,01E-01***	***1,04E+02***	***3,24E+02***	***1,05E+02***	1,55E+02	8,62E+01
	B10	***7,47E+03***	7,83E+04	***2,57E+04***	***3,15E+04***	***2,01E+04***	2,24E+05	***1,93E+06***	7,51E+05	8,48E+05	3,83E+05
	B11	***3,01E+02***	9,75E+02	***6,77E+02***	***6,72E+02***	***1,54E+02***	3,07E+02	***1,16E+03***	9,10E+02	8,71E+02	2,06E+02
	B12	1,10E+02	***2,02E+02***	1,15E+02	1,17E+02	***1,23E+01***	***1,06E+02***	1,11E+02	***1,08E+02***	***1,08E+02***	1,30E+00
	B13	***1,73E-02***	4,58E-02	***2,48E-02***	***2,62E-02***	***7,46E-03***	9,44E+01	***6,81E+03***	1,08E+02	2,43E+02	9,48E+02
	B14	4,15E+04	***4,61E+04***	4,52E+04	4,49E+04	***1,02E+03***	***3,12E+04***	3,70E+04	***3,39E+04***	***3,38E+04***	1,16E+03
	B15	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	1,75E-13	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	***0,00E+00***
TOTAL		9	4	9	10	12	8	10	8	7	3

Tableau 4-8 : Comparaison entre dynFWACM (Yu et al., 2015) et HCoCLPSO pour une dimension définie à 50 sur les problèmes *CEC'15.*

		50D									
		dynFWACM					HCoCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	B1	***8,16E+05***	***4,06E+06***	***1,67E+06***	***1,80E+06***	***7,14E+05***	1,37E+06	1,37E+06	6,44E+06	7,29E+06	4,63E+06
	B2	***7,00E-01***	***3,52E+04***	***1,30E+03***	***4,17E+03***	***8,01E+03***	9,09E+00	9,09E+00	7,68E+03	1,15E+04	1,20E+04
Multimodale	B3	***2,00E+01***	2,00E+01	***2,00E+01***	***2,00E+01***	***1,30E-06***	***2,00E+01***	2,00E+01	***2,00E+01***	***2,00E+01***	1,18E-02
	B4	1,27E+02	***4,13E+02***	2,33E+02	2,44E+02	5,88E+01	***1,15E+02***	1,15E+02	***1,89E+02***	***1,93E+02***	***4,41E+01***
	B5	4,54E+03	***7,58E+03***	5,64E+03	5,78E+03	7,00E+02	***3,21E+03***	3,21E+03	***4,69E+03***	***4,69E+03***	***6,26E+02***
Hybride	B6	***2,83E+04***	1,90E+05	***8,33E+04***	***8,81E+04***	***3,82E+04***	2,75E+05	***2,75E+05***	1,73E+06	2,28E+06	1,40E+06
	B7	1,78E+01	***1,06E+02***	5,89E+01	***4,70E+01***	2,61E+01	***1,27E+01***	1,27E+01	***4,92E+01***	***4,70E+01***	***2,23E+01***
	B8	***1,30E+04***	1,53E+05	***6,09E+04***	***6,88E+04***	***4,00E+04***	6,35E+05	***6,35E+05***	2,23E+06	2,35E+06	1,15E+06
Composition	B9	***1,03E+02***	***4,79E+02***	***1,03E+02***	***1,11E+02***	***5,26E+01***	1,06E+02	1,06E+02	1,07E+02	1,53E+02	1,00E+02
	B10	***1,35E+04***	9,26E+04	***3,84E+04***	***4,21E+04***	***1,72E+04***	1,68E+05	***1,68E+05***	1,04E+06	1,03E+06	5,03E+05
	B11	3,06E+02	***1,60E+03***	***1,11E+03***	***1,08E+03***	2,44E+02	***3,03E+02***	3,03E+02	1,36E+03	1,32E+03	***2,25E+02***
	B12	1,21E+02	***2,05E+02***	2,03E+02	1,75E+02	3,80E+01	***1,07E+02***	1,07E+02	***1,10E+02***	***1,10E+02***	***1,12E+00***
	B13	***6,43E-02***	2,58E-01	***1,08E-01***	***1,17E-01***	***4,51E-02***	1,74E+02	***1,74E+02***	1,93E+02	1,93E+02	8,27E+00
	B14	5,29E+04	***8,39E+04***	***5,33E+04***	***5,49E+04***	***6,34E+03***	***4,95E+04***	4,95E+04	5,92E+04	5,97E+04	9,75E+03
	B15	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	2,78E-13	***1,00E+02***	1,00E+02	***1,00E+02***	***1,00E+02***	***0,00E+00***
TOTAL		9	9	11	12	9	8	4	6	6	6

Tableau 4-9 : Comparaison entre dynFWACM (Yu et al., 2015) et HCoCLPSO pour une dimension définie à 100 sur les problèmes *CEC'15.*

		100D									
		dynFWACM					HCoCLPSO				
		Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type	Meil. Sol.	Pire Sol.	Méd.	Moy.	Écart-type
Unimodale	B1	***1,95E+06***	***7,50E+06***	***3,92E+06***	***3,88E+06***	***1,24E+06***	6,88E+06	6,88E+06	1,02E+07	1,04E+07	2,34E+06
	B2	***1,30E+01***	***2,00E+04***	***1,16E+03***	3,66E+03	4,63E+03	1,45E+01	1,45E+01	1,67E+03	***3,08E+03***	***3,85E+03***
Multimodale	B3	***2,00E+01***	2,00E+01	***2,00E+01***	***2,00E+01***	***2,58E-06***	***2,00E+01***	2,00E+01	***2,00E+01***	***2,00E+01***	1,69E-02
	B4	4,55E+02	***9,29E+02***	6,09E+02	6,17E+02	1,07E+02	***4,29E+02***	4,29E+02	***5,59E+02***	***5,87E+02***	***8,98E+01***
	B5	1,02E+04	***1,65E+04***	1,28E+04	1,29E+04	1,23E+03	***9,19E+03***	9,19E+03	***1,23E+04***	***1,23E+04***	***1,11E+03***
Hybride	B6	***1,90E+05***	1,43E+06	***4,43E+05***	***5,40E+05***	***2,78E+05***	1,82E+06	***1,82E+06***	4,38E+06	4,63E+06	1,83E+06
	B7	3,47E+01	***1,78E+02***	1,30E+02	1,33E+02	3,85E+01	***3,22E+01***	3,22E+01	***1,14E+02***	***1,15E+02***	***3,68E+01***
	B8	***7,72E+04***	4,32E+05	***1,63E+05***	***1,83E+05***	***7,67E+04***	5,41E+05	***5,41E+05***	2,88E+06	3,56E+06	2,03E+06
Composition	B9	1,11E+02	***1,15E+02***	1,13E+02	***1,13E+02***	***9,32E-01***	***1,09E+02***	1,09E+02	***1,11E+02***	2,86E+02	2,99E+02
	B10	1,35E+05	***4,72E+05***	2,43E+05	2,48E+05	***7,35E+04***	***3,29E+04***	3,29E+04	***2,00E+05***	***2,34E+05***	1,70E+05
	B11	***3,05E+02***	***2,90E+03***	***2,46E+03***	***2,27E+03***	6,86E+02	2,29E+03	2,29E+03	2,62E+03	2,64E+03	***1,73E+02***
	B12	1,20E+02	***2,05E+02***	1,26E+02	1,54E+02	3,84E+01	***1,07E+02***	1,15E+02	***1,18E+02***	***1,18E+02***	***1,02E+00***
	B13	***5,71E-01***	2,90E+00	***1,19E+00***	***1,36E+00***	***5,95E-01***	3,99E+02	***3,99E+02***	4,12E+02	4,12E+02	7,74E+00
	B14	1,24E+05	***1,66E+05***	1,40E+05	1,41E+05	***8,72E+03***	***1,09E+05***	1,09E+05	***1,09E+05***	***1,14E+05***	1,41E+04
	B15	1,02E+02	***1,11E+02***	1,04E+02	1,04E+02	***1,63E+00***	***1,00E+02***	1,00E+02	***1,03E+02***	***1,03E+02***	1,69E+00
TOTAL		7	11	7	7	9	9	3	9	9	6

© Hugo Deschênes

Selon les résultats des tableaux précédents, dynFWACM est la meilleure méthode de résolution pour les fonctions continues comportant une dimension définie à 10, 30 et 50. Elle a le plus souvent obtenu le plus grand nombre de meilleures solutions, de médianes, de moyennes et d'écarts-types, de même qu'un plus petit nombre de pires solutions en comparaison avec l'hybride HCoCLPSO. Cependant, le Tableau 4-9 démontre que HCoCLPSO obtient de meilleurs résultats lorsque la dimension est définie à 100. Il obtient le plus grand nombre de meilleures solutions, de médianes et de moyennes, de même que moins de pires solutions en comparaison avec dynFWACM. Il est également important de noter que l'hybride HCoCLPSO obtient de meilleurs résultats lors de la résolution de problèmes multimodaux, lesquels sont notés par les catégories *Multimodale*, *Hybride* et *Composition* dans les tableaux.

Les tableaux précédents permettent de mettre en évidence la contribution positive du HCoCLPSO pour la résolution de fonctions continues complexes de grande taille. En effet, plus la taille des problèmes augmente, plus il obtient de bons résultats sur les problèmes proposant un plus haut niveau de complexité en comparaison avec dynFWACM. L'hybride

HCoCLPSO obtient également de moins en moins de pires solutions, en passant de 11 des 15 problèmes pour une dimension définie à 10 au Tableau 4-6 pour 3 des 15 problèmes pour une dimension définie à 100 au Tableau 4-9.

4.5. CONCLUSION SUR LA CONTRIBUTION

Dans ce chapitre, l'hybridation homogène entre différentes variantes de PSO ont été élaborées puis testés sur des problèmes bien connus de nature continue, soient six fonctions classiques et quinze fonctions provenant de *CEC'15*. Trois variantes ont été utilisées pour les tests, soient le BPSO, le CLPSO et le CoLPSO. À partir de celles-ci, trois nouveaux hybrides ont été proposés, soient HCLBPSO-Half, HBPSO+CL et HCoCLPSO.

Le premier hybride, nommé HCLBPSO-Half, attribue à la moitié de la population de particules le comportement du BPSO et attribue à l'autre moitié de la population le comportement du CLPSO. La deuxième variante, nommée HBPSO+CL, ajoute le processus d'apprentissage compréhensif du CLPSO au début de la mise à jour de la population du BPSO. La troisième variante, nommée HCoCLPSO, ajoute le processus d'apprentissage compréhensif du CLPSO au début de la mise à jour d'un sous-essaim (d'une dimension) du CoLPSO.

La première partie des expérimentations de ce chapitre avait pour but de déterminer si les hybridations de PSO peuvent améliorer les variantes à partir desquelles elles sont élaborées lors de la résolution de grandes instances de problèmes continus. Les trois variantes et les trois hybrides ont été testés et comparés en utilisant six fonctions classiques continues de la littérature : *Sphere*, *Rosenbrock*, *Ackley*, *Griewank*, *Rastrigin* et *Schwefel*. Les tests effectués indiquent que l'hybride HCoCLPSO est l'hybridation offrant globalement les meilleurs résultats en comparaison avec les variantes et les autres hybridations testées pour la résolution de problèmes continus classiques, avec une dimension définie à 200. Le

HCLBPSO-Half et le HBPSO+CL offrent tout de même des performances intéressantes sur les problèmes de type unimodal, mais leurs performances diminuent considérablement avec une augmentation de la taille des instances.

La deuxième partie des expérimentations de ce chapitre avait pour but de comparer la meilleure hybridation provenant des tests précédents avec un autre algorithme de la littérature. Pour ce faire, une série de fonctions plus récentes et plus complexes, soient celles provenant de *CEC'15*, ont été utilisées. Le but de cette partie était de déterminer la capacité d'une méthode hybride conçue à partir de variantes de PSO à compétitionner avec un algorithme reconnu de la littérature. En comparant l'hybride HCoCLPSO avec dynFWACM, il est facile de remarquer une amélioration des performances avec l'utilisation d'une dimension très grande. Bien que l'hybride HCoCLPSO obtient des performances moyennes pour les dimensions 10, 30 et 50, il surpasse la capacité de dynFWACM à résoudre efficacement les problèmes avec la dimension définie à 100. Il aurait été intéressant d'observer une comparaison des résultats avec dynFWACM sur une dimension encore plus grande, telle que 200, permettant ainsi d'augmenter la complexité de la résolution et la solidité de la comparaison. Les auteurs de dynFWACM n'ont cependant pas offert des résultats avec des instances plus grandes que 100.

La capacité du CoLPSO à séparer l'espace de recherche, hybridée avec le processus d'apprentissage compréhensif du CLPSO, permet d'améliorer les solutions lors de la résolution de problèmes continus de grande taille. En effet, l'ajout du processus d'apprentissage compréhensif en début de mise à jour d'un sous-essaim permet de poursuivre une exploration efficace de l'espace de recherche lorsque les solutions ne permettent pas de contribuer positivement à la meilleure solution cognitive. Ces deux éléments, c'est-à-dire la séparation de l'espace de recherche en sous-essaims et l'ajout d'un processus d'apprentissage compréhensif, forme une stratégie de conception efficace. L'hybridation ainsi formée, nommée HCoCLPSO, permet de valider le premier objectif

secondaire, qui est de *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature*.

CHAPITRE 5 – PROPOSITION DE DISCRÉTISATIONS DU PSO POUR L'OPTIMISATION COMBINATOIRE

Cette section a fait l'objet d'un acte de conférence :

Deschênes H., Gagné C., *Discrétisation de l'optimisation par essaim particulaire sur une base commune*, 11th International Conference on Modeling, Optimization & Simulation (MOSIM 2016), Montréal, Canada, 22-24 August 2016

5.1. INTRODUCTION

Ce chapitre aborde la résolution de problèmes combinatoires par l'utilisation du PSO. Tel que mentionné précédemment, une métaheuristique doit être adaptée à un problème d'optimisation afin qu'elle soit performante lors de la résolution de problèmes NP-difficiles. La discrétisation est la deuxième stratégie de conception abordée dans cette thèse. Elle permet d'accroitre la capacité de résolution du PSO, et cette fois-ci, pour la résolution de problèmes comportant des variables de décision de nature entière.

Les procédés de discrétisation présentés dans ce chapitre proviennent de la littérature. Ils sont toutefois difficilement comparables étant donné qu'ils ont été expérimentés dans des conditions différentes et sur des problèmes différents. En les ramenant sur une base commune, il sera possible d'en dégager les avantages et les inconvénients de chacun, de même que les conditions qui les rendent pertinents pour la résolution de problèmes combinatoires.

Les sections suivantes sont organisées comme suit. Tout d'abord, la Section 5.2 présente deux problèmes combinatoires à la base des expérimentations de ce chapitre, soient le SOP et le RCPSP. Par la suite, la Section 5.3 détaille les trois procédés de discrétisation du PSO sélectionnés pour résoudre ces deux problèmes : selon les priorités d'ordonnancement (H. Zhang et al., 2006), par listes de translations (Anghinolfi & Paolucci, 2009) et par listes de permutations (Clerc, 2004). Finalement, la Section 5.4 présente les détails expérimentaux permettant de comparer les procédés de discrétisation énumérés sur une base commune selon la qualité des solutions obtenues et les temps d'exécution.

5.2. PRÉSENTATION DES PROBLÈMES DE SUPPORT

Le problème d'ordonnancement séquentiel (SOP) consiste à ordonner les villes d'un parcours de manière à minimiser le coût total (Escudero, 1988). L'ordre des visites au sein d'une solution est important considérant que ce problème modélise des contraintes de préséance. Il s'agit d'une version alternative au problème asymétrique du voyageur de commerce.

Le but du SOP est d'ordonnancer un ensemble de K villes possédant des contraintes de préséance. L'ajout d'une ville au sein du vecteur solution n'est valide que si ses préséances sont déjà ordonnancées au préalable. Chaque paire de villes est accompagnée d'une distance unique servant au calcul de la fonction objectif. Le but du SOP est de minimiser la distance nécessaire afin de parcourir l'ensemble des K villes du problème.

Le problème d'ordonnancement de projets avec contraintes de ressources (RCPSP) consiste à ordonnancer une série d'activités en prenant en compte des contraintes typiques de la gestion de projets : une durée d'exécution, le respect de la préséance entre des activités et le respect de la quantité de ressources nécessaires pour compléter les activités du projet (Kolisch & Sprecher, 1997).

Le but du RCPSP est d'ordonnancer un ensemble de J activités où chacune d'entre elles doit être traitée selon un nombre défini de contraintes. En plus d'utiliser des contraintes de préséances entre les différentes activités, il y a une gestion nécessaire des ressources disponibles. Chaque activité doit satisfaire les demandes selon les R ressources différentes du problème et peut être exécutée lorsque ses activités préalables sont terminées. Chaque activité a donc une durée de traitement, des contraintes de préséance et un nombre précis de ressources nécessaires à son exécution. Les activités 0 et $J + 1$ correspondent au début et à la fin du projet. Elles possèdent une durée nulle et ne consomment aucune ressource à

leur exécution. L'objectif à atteindre pour le RCPSP est de minimiser la durée totale du projet, tout en assurant un respect des différentes contraintes du projet. Ce problème propose un niveau plus élevé de contraintes que le SOP et en augmente alors la complexité de la résolution.

Les instances des problèmes SOP et RCPSP peuvent être représentés de la même manière par un vecteur solution q. Ce vecteur est composé de valeurs entières représentant les éléments à ordonnancer (les villes pour le SOP ou les activités pour le RCPSP). Ce vecteur d'éléments doit respecter les contraintes imposées par les préséances et par la quantité de ressources afin qu'il soit considéré valide pour l'évaluation par la fonction objectif. Par l'utilisation du PSO, le vecteur réel composant la position x_i d'une particule contient une solution au problème qui doit être discrétisée afin de correspondre au format requis par le vecteur solution q. Les procédés de discrétisation décrits à la section suivante transforment donc la position x_i en solution q afin que les valeurs traitées prennent un sens dans le cadre du problème à résoudre.

5.3. ADAPTATION DES PROCÉDÉS DE DISCRÉTISATION

Trois procédés de discrétisation provenant de ceux présentés à la Section 3.4 sont détaillés dans cette section. Ils ont été sélectionnés en raison de leur nature polyvalente et des performances qu'elles démontrent pour la résolution de problèmes, selon leurs auteurs respectifs. Le premier procédé de discrétisation, par priorité d'ordonnancement, est proposé par H. Zhang et al. (2005). Il fait partie des procédés de la catégorie *Manipulation de nombres réels* (Tableau 3-1). Les deux autres procédés font partie de la catégorie *Manipulation de listes de nombres entiers* (Tableau 3-1). Le deuxième procédé de discrétisation, par liste de translations, est proposé par Anghinolfi et Paolucci (2009). Le troisième procédé de discrétisation, par liste de permutations, est proposé par Clerc (2004).

Les détails de leur conception utilisent les redéfinitions des opérateurs arithmétiques présentés à la Section 3.4 et sont redéfinis en détail dans la Section 5.3.2.

Il est important de noter que tous les procédés de discrétisation présentés dans ce chapitre garantissent la permutation des villes du SOP et des activités du RCPSP. Cependant, ils ne garantissent pas le respect des préséances pour le SOP et le RCPSP, ni le respect des contraintes de ressources pour le RCPSP. Ceci implique que toute solution résultant de la discrétisation doit être vérifiée afin de valider l'ordonnancement des éléments. Une solution ne respectant pas les contraintes de préséance et de ressources devra subir potentiellement une transformation afin de produire une solution valide en vue de l'évaluation par la fonction objectif. Les sections qui suivent présentent les trois procédés de discrétisation avec la redéfinition des opérateurs arithmétiques, suivi des processus de réparation des solutions.

5.3.1. PREMIER PROCÉDÉ : PAR PRIORITÉ D'ORDONNANCEMENT

La discrétisation par priorité d'ordonnancement proposée par H. Zhang et al. (2005) fait partie de la catégorie *Manipulation de nombres réels*. Cette discrétisation utilise le tri décroissant de nombres réels afin d'ordonnancer une série d'éléments dénombrables. Il y a utilisation d'une position x_i^t composée de nombres contraints dans l'ensemble des nombres réels entre 0 et 1. Les valeurs contenues dans la position d'une particule représentent ainsi des priorités d'ordonnancement. La vélocité v_i^t est également composée de nombres réels permettant de varier les priorités contenues dans la position. Ces nombres sont contraints dans l'ensemble des nombres réels entre -1 et 1. En raison de la nature de la composition de la position et de la vélocité, aucune redéfinition des opérateurs arithmétiques n'est nécessaire. Cette discrétisation est très similaire au quatrième procédé proposé par Krause et al. (2013) en permettant également de transformer un vecteur composé de nombres réels en nombres entiers.

La spécificité de cette discrétisation concerne l'utilisation d'une fonction de transformation. Elle est utilisée afin de trier le contenu de la position x_i d'une particule i à l'itération t. Une valeur élevée pour une variable du vecteur position signifie que la dimension (l'emplacement) où est située ladite valeur est l'élément inscrit en premier dans le vecteur solution q_i^t. La plus petite valeur est donc la dernière à être ajoutée au vecteur solution. Par exemple, une position $x_i^t = (0.58, 0.65, 0.33, 0.77, 0.09, 0.39)$ pour un problème comportant six variables à ordonnancer permet d'obtenir la solution $q_i^t = (4, 2, 1, 6, 3, 5)$. La plus grande valeur de la position (0.77) est située à la dimension 4. La valeur 4 est donc le premier élément ordonnancé. 0.09 correspond à la plus petite valeur retrouvée dans la position. Celle-ci est située à la dimension 5, signifiant que cette variable est la dernière à être ordonnancée et elle est ajoutée au vecteur solution en dernier.

Ce procédé de discrétisation est utilisé, entre autres, afin de résoudre le RCPSP (R.-M. Chen et al., 2010; H. Zhang et al., 2005). Une version alternative utilisant une matrice de nombres réels au lieu d'un vecteur pour la composition de la position et de la vélocité d'une particule est proposée par d'autres auteurs afin de résoudre le problème du voyageur de commerce et le problème d'ordonnancement d'ateliers (Pang et al., 2004; Sha & Hsu, 2006). Elle est également utilisée pour résoudre le problème de planification des travailleurs à domicile (Akjiratikarl et al., 2007).

5.3.2. RAPPEL DES OPÉRATIONS ARITHMÉTIQUES

Tel que mentionné à la Section 3.4, les opérateurs arithmétiques doivent être redéfinis lorsque la nature des variables de décision n'appartient plus au domaine des nombres réels. Cette section vise à démystifier ces changements avant de décrire les deux prochains procédés de discrétisation.

Tout d'abord, rappelons les équations de mise à jour d'une particule tel que présentées à la Section 3.2.1. Cette fois, les opérateurs arithmétiques sont mis en évidence dans l'Équation 3-1 et l'Équation 3-2 avec les lettres *OP-1*, *OP-2*, *OP-3* et *OP-4*.

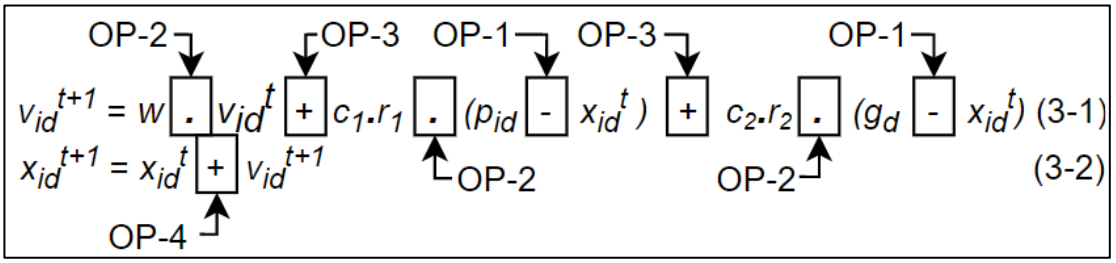

Figure 5-1 : Mise en évidence des opérateurs arithmétiques provenant de l'Équation 3-1 et de l'Équation 3-2 de la mise à jour d'une particule du PSO.
© Hugo Deschênes

Dans les équations ci-haut, il est possible de distinguer quatre types d'opérateurs arithmétiques à redéfinir dans le cadre d'une discrétisation :

OP-1 La soustraction $(-)$ entre deux vecteurs position $((p_i - x_i^t)$ et $(g - x_i^t))$ permet d'obtenir un vecteur vélocité représentant la distance entre la position x_i^t et p_i ou g ;

OP-2 La multiplication $(\cdot)$ d'un scalaire (w, $c_1 \cdot r_1$ et $c_2 \cdot r_2$) par une vélocité, résultant en un nouveau vecteur vélocité ;

OP-3 L'addition $(+)$ entre plusieurs vélocités permettant de produire la vélocité résultante v_i^{t+1} pour l'itération t. Cette addition concerne les trois résultats provenant de la multiplication d'un vecteur vélocité avec un scalaire de l'opérateur OP-2 ; et

OP-4 L'addition $(+)$ entre une position x_i^t et une vélocité v_i^{t+1}, résultant en une nouvelle position x_i^{t+1} au sein de l'espace de recherche.

5.3.3. DEUXIÈME PROCÉDÉ : PAR LISTE DE TRANSLATIONS

Cette discrétisation s'inscrit dans la catégorie *Manipulation de listes de nombres entiers*. Elle apporte un changement au déplacement des particules par l'application d'une série de translations des variables composants le vecteur position de chacune des particules du PSO. Elle utilise une position x_i composée d'une liste de nombres entiers à ordonnancer. La vélocité v_i est composée d'une série de translations, nommées pseudo-insertions (PI), lesquelles sont également composées de nombres entiers (Anghinolfi & Paolucci, 2009). Ces translations définissent le déplacement à effectuer à chacune des variables au sein du vecteur position.

Une PI est identifiée par une paire $y = (a, \lambda)$, où a représente la variable de décision provenant de l'ensemble A des variables composant le problème combinatoire. λ représente le déplacement (la translation) en valeur entière à effectuer à une variable a au sein du vecteur position d'une particule. Il ne s'agit pas du déplacement au sein de l'espace de recherche, mais de la translation de l'élément composant la solution à un autre emplacement du vecteur solution. La composition de la vélocité est définie par l'Équation 5-1, où Y représente le nombre total de PI à appliquer et y la PI en traitement.

$$v_{id} = \{(a, \lambda)_y \mid y = 1, \dots, Y; a \in A\} \quad (5\text{-}1)$$

Appliquer un ensemble de PI à la position d'une particule équivaut à ajouter la vélocité d'une particule à sa position comme dans l'Équation 3-2

(3-2. Il s'agit de la redéfinition de l'opérateur de *somme entre une vélocité et une position*. Il est possible que le résultat de cette opération produise une situation de conflit entre deux variables se retrouvant au même emplacement. En effet, deux variables de décision ne peuvent pas être contenues à la même dimension dans le vecteur

solution. Une dimension ne peut contenir qu'une seule valeur. Un processus d'ajustement est appliqué pour corriger cette situation problématique.

Afin de bien visualiser les concepts, utilisons l'exemple d'Anghinolfi et Paolucci (2009) : soit une particule i à une itération t définie par la position $x_i^t = (1, 2, 3, 4)$. Posons également la vélocité $v_i^t = \big((1,0), (2,2), (3, -2), (4,0)\big)$ de cette même particule, contenant alors quatre PI. Au sein de cette vélocité, la première PI contient la variable 1 et n'est associée à aucun déplacement. La variable 2 a un déplacement de 2 dimensions vers la droite, la variable 3 a un déplacement de deux dimensions vers la gauche et aucun déplacement n'est appliqué à la variable 4. Lorsque cette vélocité est additionnée à la position x_i^t avec l'Équation 3-2 de la mise à jour de la position d'une particule, la variable 2 rejoint la dimension de la variable 4 et la variable 3 rejoint la dimension de la variable 1. Cette opération produit la position $x_i^{t\prime} = (1\ 3, -, -, 4\ 2)$. Deux variables ne peuvent coexister au même emplacement et il y a donc une situation de conflit. Le processus correctif élaboré par Anghinolfi et Paolucci (2009) consiste à parcourir tous les éléments de la position et de détecter les deux anomalies suivantes :

- Plus d'une variable est localisée à la même dimension de la position $x_1^{t\prime}$; et
- Une des dimensions de la position $x_1^{t\prime}$ est vide.

Pour expliquer la procédure de correction, reprenons l'exemple précédent. La première anomalie rencontrée au sein de $x_1^{t\prime}$ est située à la première dimension. Les variables 1 et 3 s'y retrouvent dans cet ordre. La première variable (1) de ce conflit doit être reportée à la dimension suivante, qu'elle soit déjà occupée par une autre variable ou pas. Les conflits à la première dimension étant réglés, la deuxième dimension est analysée. Cette dimension ne contient désormais qu'une seule variable (1) et ne nécessite aucune correction. La troisième dimension est cependant vide. La prochaine variable trouvée dans les dimensions subséquentes est sélectionnée et mise à cet emplacement. Dans l'exemple utilisé, les variables subséquentes sont 4 et 2. La première des deux variables étant 4, celle-ci est devancée à la troisième dimension afin de combler le vide repéré. La quatrième dimension

ne contient désormais qu'une seule variable et ne nécessite aucun ajustement. Le résultat de l'application de la vélocité sur la position est donc $\mathrm{x}_i^{\mathrm{t}+1} = (3, 1, 4, 2)$.

Il est possible de déduire que la différence entre la position de deux particules peut être représentée par une série de PI. Cette différence se produit lors de la mise à jour de la vélocité d'une particule au sein de l'Équation 3-1. Il s'agit de la redéfinition de l'opérateur arithmétique de *soustraction entre deux positions*. Par exemple, soient la position d'une particule $x_i^t = (1, 2, 3, 4)$ et la position de la meilleure solution globale $g^t = (2, 3, 1, 4)$. Le calcul de la soustraction $g^t - x_i^t$, telle qu'elle est retrouvée dans l'Équation 3-1, permet d'obtenir la liste de PI permettant à x_i^t de rejoindre l'emplacement de g^t. Il s'agit de la distance entre les deux positions, où $g^t - x_i^t = \big((1,2), (2,-1), (3,-1), (4,0)\big)$.

Une autre modification aux opérateurs arithmétiques est l'*addition de deux vélocités*, résultant en une troisième vélocité combinant les deux premières. Cette opération se retrouve également dans l'Équation 3-1. Il y a addition de la valeur de déplacement des variables identiques provenant des deux vélocités. Par exemple, posons une première vélocité $v_{i_1}^t = \big((1,0), (2,2), (3,-2), (4,0)\big)$ et une deuxième vélocité $v_{i_2}^t = \big((1,3), (2,1), (3,2), (4,-1)\big)$. En additionnant les déplacements λ des variables identiques, la vélocité résultante devient $v_{i_2}^t + v_{i_1}^t = \big((1,3), (2,3), (3,0), (4,-1)\big)$.

La *multiplication entre un nombre réel et une vélocité* est également ajustée. Tel que défini précédemment, une PI est composée d'une variable et de la valeur de son déplacement. La multiplication d'un nombre réel par une vélocité consiste donc en la multiplication de ce nombre par la valeur du déplacement. Ainsi, un facteur $c_1 = 1.5$ multiplié par une vélocité $v_i^t = \big((1,0), (2,2), (3,-3), (4,0)\big)$ résulte en une nouvelle vélocité $v_i^{t\prime} = \big((1,0), (2,3), (3,-5), (4,0)\big)$. Étant donné que λ doit être une valeur discrète, et que la multiplication par des nombres réels peut résulter en un nombre réel, il y a un arrondissement de la valeur résultante. La sélection entre le plancher et le plafond de

l'arrondi est aléatoire lors de la multiplication de c_1 par λ. Également, l'emplacement résultant d'un déplacement λ ne peut pas être inférieur à la valeur zéro ni supérieur à la dimension D d'une particule. Il y a donc constriction du résultat obtenu à l'intérieur de ces valeurs.

Lors de l'initialisation de ce procédé de discrétisation, un nombre aléatoire de PI est généré. Anghinolfi et Paolucci (2009) utilisent un nombre aléatoire de PI variant entre $N/4$ et $N/2$ pour chacune des particules, où N représente le nombre total de particules de l'essaim. La valeur de λ est quant à elle générée aléatoirement entre $-N/3$ et $N/3$ en assurant le respect des bornes du problème.

Ce procédé de discrétisation est utilisé pour la résolution du problème d'ordonnancement d'une machine unique avec réglages dépendants de la séquence des travaux et minimisation du retard total pondéré (*single-machine total weighted tardiness scheduling with sequence-dependent setup times problem*) (Anghinolfi & Paolucci, 2009), du problème d'ordonnancement séquentiel (*sequential ordering problem*) (Anghinolfi et al., 2011) et du problème d'ordonnancement de projets avec contraintes de ressources (*resource constrained project scheduling problem*) (Lemamou et al., 2010). Une version alternative de ce procédé est également utilisée pour la résolution du problème du voyageur de commerce (*traveling salesman problem*) (X. Wang et al., 2013).

5.3.4. TROISIÈME PROCÉDÉ : PAR LISTE DE PERMUTATIONS

L'utilisation de translations n'est pas le seul type de mouvement qu'il est possible d'appliquer aux éléments composant la position d'une particule. Le procédé de discrétisation décrit dans cette section utilise la permutation de variables afin de déplacer les particules de l'essaim (Clerc, 2004). Elle fait également partie de la catégorie *Manipulation de listes de nombres entiers*. La position x_i d'une particule est composée de nombres entiers et sa vélocité v_i est composée d'une liste de permutations de nombres entiers. Cette vélocité est

représentée à l'aide de l'Équation 5-2, où il y a une suite de permutations entre deux variables au sein de la position d'une particule. Dans cette Équation, α_1 et α_2 représentent deux variables permutables provenant de l'ensemble A des variables de décision du problème, U représente le nombre total de permutations u contenues dans la vélocité.

$$\mathrm{v_i^t} = \{(a_1, a_2)_u, u = 1, \dots, U;\ a_1, a_2 \in A\} \tag{5-2}$$

Soit une particule comportant une position $x_i^t = (1, 2, 3, 4, 5)$ et une vélocité $v_i^t = \big((1, 2), (2, 3)\big)$. Cette vélocité contient un total de deux permutations, soient un échange entre les variables 1 et 2, suivi d'un échange entre les variables 2 et 3. Appliquer cette vélocité à la position d'une particule équivaut à redéfinir l'opérateur de *somme entre une vélocité et une position*. En appliquant ces deux permutations successivement à x_i^t, la position de la particule devient tout d'abord $x_i^{t\prime} = (2, 1, 3, 4, 5)$, puis $x_i^{t\prime\prime} = (3, 1, 2, 4, 5)$. Les permutations ne sont pas commutatives. Elles doivent obligatoirement être appliquées dans l'ordre qu'elles sont indiquées dans la vélocité, sans quoi le résultat ne serait pas constant.

Considérant qu'il est possible d'additionner une liste de permutations à une position afin d'obtenir une nouvelle position, il est possible d'effectuer l'opération inverse : *soustraire deux positions* afin d'obtenir une liste de permutations. Par exemple, une soustraction entre la position d'une particule $x_i^t = (1,2,3,4,5)$ et la meilleure position collective $g = (2,5,1,4,3)$ permet d'obtenir la vélocité $v_i^{t+1} = \big((1, 2), (2, 5), (3, 5)\big)$. Cette vélocité est obtenue en appliquant successivement des permutations à x_i^t afin d'obtenir g. Si $x_i^t + v_i^t = g$, alors ceci implique que $g - x_i^t = v_i^t$. Il est important de noter que le résultat de la soustraction n'est pas une solution unique. Il est donc possible d'utiliser une liste différente de permutations permettant d'obtenir les mêmes résultats, par l'utilisation d'une vélocité alternative $v_i^{t\prime} = \big((3,5), (1,3), (1,2)\big)$. La différence entre ces deux vélocités est obtenue lorsque les permutations sont construites en débutant par la première dimension de la position ou en

débutant par la dernière dimension. L'important est d'être constant pour la direction sélectionnée et de s'y tenir pour toute la résolution du problème (Clerc, 2004).

Les valeurs composant la vélocité peuvent être négatives lorsqu'elles font partie de l'ensemble des nombres réels. Un déplacement doit permettre l'utilisation de nombres négatifs afin de représenter une direction. Contrairement à la discrétisation par liste de translations de la section précédente, le concept de déplacement positif et négatif dans le cas d'une série de permutations n'existe pas. Clerc (2004) propose un moyen de calculer le négatif d'un déplacement par l'inversion des éléments à soustraire. Au lieu d'effectuer la soustraction $g - x_i^t = v_i^t$, le calcul est remplacé par $x_i^t - g = \neg v_i^t$.

L'*addition de deux vélocités*, résultant en une troisième vélocité combinant les deux premières, peut être effectuée. Pour ce faire, le vecteur résultant est constitué d'une combinaison des permutations provenant des deux vélocités selon l'ordre de leur apparition dans l'addition. Par exemple, l'addition entre $v_{i_1}^t = \big((1,2),(2,4)\big)$ et $v_{i_2}^t = \big((3,5),(1,2)\big)$ résulterait en une nouvelle vélocité $v_{i_1}^t + v_{i_2}^t = \big((1,2),(2,4),(3,5),(1,2)\big)$.

Une autre redéfinition d'opérations concerne la *multiplication d'un nombre scalaire par une vélocité*. Considérant l'utilisation non conventionnelle du concept de « distance » dans une permutation, il est impossible d'utiliser une valeur réelle afin de réduire un déplacement. Pour corriger ce problème, il y a une augmentation ou une diminution du nombre total de permutations composant la vélocité selon la valeur du coefficient. Quatre cas sont distingués par Clerc (2004) sur cette multiplication. Le premier cas est obtenu lorsque le coefficient est nul (0). La vélocité est alors vidée de toute permutation qu'elle contient. Le deuxième cas est obtenu lorsque la valeur du coefficient est entre 0 et 1. La vélocité est alors tronquée selon la même proportion que la valeur du coefficient. Posons par exemple une vélocité $\mathrm{v_i^t} = \big((1,2),(3,7),(2,5),(8,9),(3,4)\big)$ et un coefficient $c_1 = 0.7$. La valeur de c_1 signifie que 70% de la vélocité est conservée après la multiplication. Seulement les trois premières permutations sont conservées après la multiplication puisqu'il s'agit de la portion entière du

produit entre le coefficient (0.7) et le nombre de permutations (5). Les décimales résultantes de cette dernière multiplication sont tronquées, résultant en la nouvelle vélocité $\mathrm{v'}_{\mathrm{i}}^{\mathrm{t}} = \big((1,2),(3,7),(2,5)\big)$. Le troisième cas est obtenu lorsque le coefficient est plus grand que 1. Il est alors nécessaire d'effectuer le calcul en trois étapes :

1- La vélocité est additionnée à elle-même autant de fois que la partie entière du coefficient ;
2- La partie fractionnaire est traitée de manière identique au deuxième cas décrit ci-haut ; et
3- Il y a addition entre le résultat de la première étape et celui de la deuxième étape en ajoutant les permutations les unes à la suite des autres.

Finalement, le quatrième cas concerne tout coefficient plus petit que 0. Le traitement à effectuer consiste à traiter la valeur du coefficient en valeur absolue et d'inverser le calcul de la vélocité. Ainsi, $v_i^t = |c_1|\ (\neg v_i^t)$.

L'implémentation de cette discrétisation peut cependant mener à une explosion du nombre de permutations au sein d'une même vélocité. Certaines combinaisons de permutations peuvent s'annuler s'il y a plus de permutations que le nombre total de dimensions D d'une particule. Un nombre supérieur à D implique qu'au moins une variable est déplacée à deux reprises, causant alors une ou plusieurs permutations superflues et augmentant les temps de calculs inutilement.

Lors de l'initialisation de ce procédé de discrétisation, un nombre aléatoire de permutations est généré. L'auteur utilise un nombre variant entre $N/4$ et $N/2$ pour chacune des particules, où N représente le nombre total de particules dans l'essaim.

Cette discrétisation est utilisée pour la résolution du problème du voyageur de commerce (*traveling salesman problem*) (Clerc, 2004). Elle est également utilisée pour la résolution du problème d'ordonnancement d'ateliers en série (*flowshop scheduling problem*)

(uni et multi-objectif) (Ponnambalam et al., 2009). Zhong et Lei (2010), ainsi que X. H. Shi et al. (2007), utilisent quant à eux une version alternative de cette discrétisation afin de résoudre eux aussi le problème du voyageur de commerce.

5.3.5. RÉPARATION DE SOLUTIONS

Tel que mentionné précédemment, la discrétisation prend en considération la permutation des éléments à ordonnancer, mais ne prend pas en considération le respect des préséances et des ressources. La réparation des solutions est importante afin d'être en mesure d'évaluer une solution réalisable avec la fonction objectif.

Pour le SOP, le processus de réparation d'une solution parcours les éléments ordonnancés. Ce processus valide que chaque ville ordonnancée respecte les contraintes de préséances. Si une ville ne peut pas être ordonnancée à un emplacement étant donné qu'elle brise les règles de préséances, elle est mise dans une file d'attente. Le parcours de la solution suit son cours. La file d'attente demeure en observation pendant ce temps. Dès que les contraintes de préséance d'une ville sont respectées, ladite ville est ordonnancée au sein du vecteur solution. Si plusieurs villes sont admissibles, la première entrée dans la file est ordonnancée en premier, et ainsi de suite.

Le RCPSP suit le même principe que le SOP pour le respect des préséances. Cependant, des contraintes additionnelles de respect de l'utilisation des ressources peuvent empêcher une activité d'être ordonnancée dès que la préséance est respectée. Pour corriger la situation, l'activité respectant les préséances demeure dans la file d'attente et sera ordonnancée dès que toutes les ressources sont disponibles pour son exécution.

5.4. EXPÉRIMENTATIONS NUMÉRIQUES

Les tests effectués dans cette section visent à identifier, de manière générale, quel procédé de discrétisation propose de meilleurs résultats sur une grande étendue d'instances à complexité variable. Considérant qu'il est toujours nécessaire d'adapter une métaheuristique à un problème à résoudre pour que la méthode devienne performante et pertinente, cette section se concentre sur l'apport positif des procédés de discrétisation, et la manière qu'ils contribuent à améliorer les stratégies métaheuristiques. Plutôt que de viser un très grand nombre d'exécutions sur un nombre réduit d'instances, la stratégie employée dans cette section consiste à résoudre un plus grand nombre d'instances. La principale caractéristique recherchée avec cette approche est de valider le procédé de discrétisation possédant la performance moyenne la plus robuste. Ces procédés sont utilisés dans leur forme la plus épurée sans amélioration complémentaire, tout en conservant les processus de réparation des solutions pour la résolution du SOP et du RCPSP.

Les instances de test utilisées pour résoudre le SOP sont fournies par TSPLIB. Un total de 41 instances est disponible en téléchargement, incluant l'information concernant les meilleures solutions connues pour ces instances. Elles sont disponibles à l'adresse suivante : http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

Les instances tests utilisées pour résoudre le RCPSP se divisent en trois catégories selon le nombre d'activités : 30, 60 et 120 (nommées respectivement J30, J60 et J120). Les ensembles J30 et J60 comportent chacune 480 instances, alors que l'ensemble J120 en comporte 600. Les solutions optimales pour l'ensemble J30 sont disponibles. Cependant, cette information n'est pas connue pour les autres ensembles (J60 et J120). Les meilleures solutions connues sont mises à jour sur le site web de PSPLIB. L'ensemble des instances est disponible à l'adresse suivante : http://www.om-db.wi.tum.de/psplib/data.html.

5.4.1. DÉFINITION DES PARAMÈTRES D'EXÉCUTION

Les tests sont exécutés sur un ordinateur comportant le système d'exploitation Windows 10 Professionnel, avec un processeur Intel® Core™ i5-2400 incluant un CPU à 3.10GHz et 8,00 Go de mémoire vive. Les paramètres du PSO sont fixés selon les auteurs des procédés de discrétisation, et ce, pour chacun des problèmes de support. Les paramètres de confiance cognitive c_1 et de confiance collective c_2, de même que le paramètre d'inertie w, sont définis au Tableau 5-1. Le critère déterminé d'arrêt de chaque exécution est fixé à 50 000 évaluations de solutions par la fonction objectif. Le nombre total de particules N est défini à 50. La valeur des paramètres est déterminée afin de permettre à la fois une grande couverture du problème à l'étude avec 1 000 évaluations par particule. Au total, cinq exécutions sont effectuées pour chacune des instances.

Tableau 5-1 : Paramètres définis par les auteurs des procédés de discrétisation pour l'exécution sur les problèmes SOP et RCPSP.

Procédés de discrétisation	w	c_1	c_2
Priorité d'ordonnancement	0,5	1,5	2
Liste de translations	0,5	1,5	1,5
Liste de permutations	1	2	2

5.4.2. PRÉSENTATION DES RÉSULTATS

Les tableaux suivants présentent les résultats obtenus avec l'utilisation des trois procédés de discrétisation décrits précédemment. Les résultats du SOP sont présentés en premier étant donné qu'il s'agit du problème présentant un niveau de contraintes moins élevé. Les résultats du RCPSP sont présentés par la suite, avec la résolution des instances J30, J60 et J120. Il est important de noter que les résultats obtenus dans cette section ne cherchent pas à compétitionner avec des algorithmes de la littérature. Les procédés de

discrétisation sont utilisés sans processus complémentaires permettant d'améliorer davantage les solutions obtenues et sont utilisés dans leur forme la plus épurée.

5.4.2.1. PREMIER PROBLÈME DE SUPPORT : SOP

Le Tableau 5-2 montre les résultats obtenus pour la résolution du SOP. Chacune des 41 instances est testée à cinq reprises. La moyenne générale des résultats par exécution est utilisée pour la comparaison afin d'avoir une idée globale de la performance des procédés de discrétisation. Le tableau présente ainsi la déviation moyenne et la déviation maximale des meilleures solutions connues, de même que le pourcentage des meilleures solutions trouvées par exécution. La moyenne générale pour chacune des statistiques est ajoutée à la dernière ligne du tableau.

Tableau 5-2 : Moyennes des déviations et déviation maximale pour la résolution des 41 instances de SOP.

	Priorité d'ordonnancement			Liste de translations			Liste de permutations		
Exéc.	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé
1	161,45%	1756,25%	4,88%	138,32%	1432,13%	4,88%	154,69%	1805,96%	4,88%
2	159,19%	1775,68%	2,44%	131,79%	1419,04%	7,32%	152,54%	1709,57%	4,88%
3	164,13%	1733,20%	4,88%	135,80%	1352,73%	4,88%	155,69%	1778,03%	7,32%
4	155,69%	1664,36%	2,44%	134,79%	1317,19%	4,88%	151,75%	1758,01%	4,88%
5	157,72%	1709,96%	2,44%	137,65%	1498,83%	4,88%	154,00%	1704,00%	4,88%
Moy.	**159,64%**	**1727,89%**	**3,41%**	**135,67%**	**1403,98%**	**5,37%**	**153,74%**	**1751,11%**	**5,37%**

Les résultats pour la résolution des 41 instances de SOP permettent de tirer des conclusions intéressantes. Le Tableau 5-2 démontre la capacité qu'a le procédé de discrétisation par liste de translations à résoudre de manière plus efficace les instances comparativement aux deux autres procédés de discrétisation. En effet, son taux de déviation moyen de 135,67% est inférieur à celui des deux autres procédés de discrétisation, lesquels obtiennent une déviation moyenne de 159,64% pour la priorité d'ordonnancement et 153,74% pour la liste de permutations. Également, le procédé de discrétisation par liste de translations offre le plus bas taux de la moyenne des déviations maximales obtenues pour

l'ensemble des instances. Ceci signifie qu'il s'agit du procédé proposant une meilleure constance en termes de résultats obtenus sur une grande variété d'instances. Concernant la troisième statistique, c'est-à-dire le pourcentage de meilleures solutions trouvées, l'utilisation des priorités d'ordonnancement propose en moyenne moins souvent des solutions correspondant à la meilleure solution connue avec 3,41% des solutions obtenues. Sur cet élément, les deux autres procédés de discrétisation (par liste de translations et par liste de permutations) sont à égalités avec 5,37% des meilleures solutions obtenues.

La qualité de solution n'est pas le seul facteur de comparaison à prendre en considération afin de valider le meilleur procédé de discrétisation. La Figure 5-2 montre la moyenne des temps d'exécution selon la taille des instances des problèmes testés, c'est-à-dire, selon le nombre de villes.

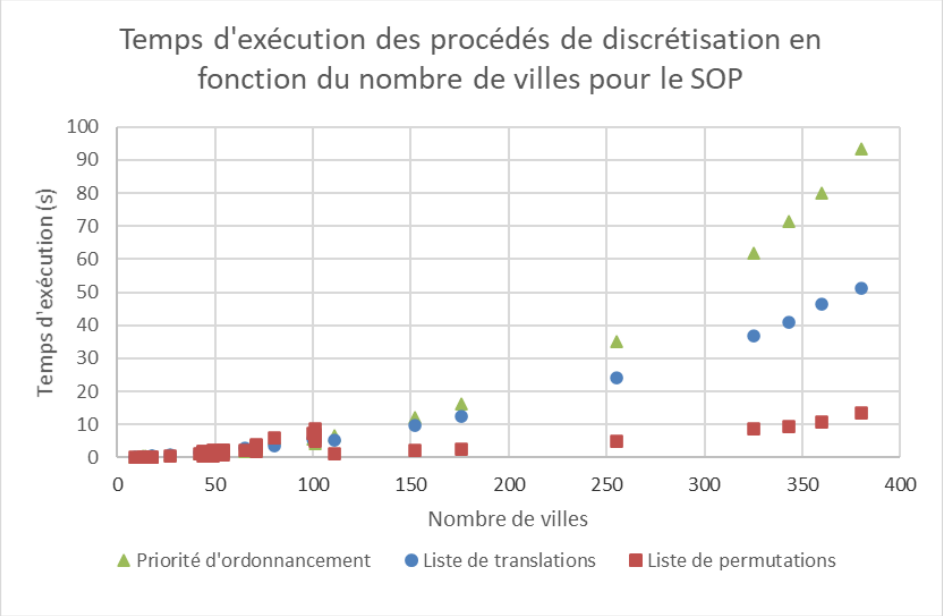

Figure 5-2 : Moyenne des temps d'exécution en secondes lors de la résolution du SOP avec les procédés de discrétisation selon le nombre de villes des instances.
© Hugo Deschênes

Avec cette perspective des résultats, il est possible d'observer que le temps de résolution n'est pas linéaire. Il semble augmenter de façon exponentielle selon le nombre de

villes utilisées. Le procédé de discrétisation proposant une plus faible pente et les temps d'exécution les plus bas correspondent à la liste de permutations, suivi en deuxième avec la liste de translations et en troisième avec les priorités d'ordonnancement.

La différence entre les temps d'exécution peut s'expliquer par différents facteurs. Tout d'abord, la vélocité des particules pour le procédé de discrétisation par la liste de permutations possède une longueur variable. Ceci implique que moins d'opérations peuvent être effectuées pour déplacer une particule, où le nombre de permutations peut varier entre 1 et D, la dimension du problème. À l'opposé, le procédé par priorité d'ordonnancement nécessite davantage de recherche à l'intérieur du vecteur position afin d'ordonnancer les activités selon les priorités en valeurs réelles qui y sont inscrites. Ceci peut impliquer que le vecteur position est parcouru, dans le pire cas, $D - 1$ fois afin de trouver la prochaine valeur à ordonnancer. Dans le milieu, le procédé par liste de translations est le plus constant considérant qu'il y aura toujours D translations à effectuer, avec des ajustements mineurs si deux activités se retrouvent au même endroit.

5.4.2.2. DEUXIÈME PROBLÈME DE SUPPORT : RCPSP

Les résultats de la résolution du RCPSP sont également comparés par l'utilisation de la déviation moyenne et la déviation maximale selon les solutions optimales pour les instances J30, et selon les meilleures solutions connues pour les instances J60 et J120. Le pourcentage des solutions atteintes est également présenté. Chacun des tableaux suivants montre la moyenne des résultats par exécution avec la moyenne globale ajoutée à la dernière ligne. Le Tableau 5-3 montre les résultats pour les 480 instances J30. Le Tableau 5-4 pour les 480 instances J60, et le Tableau 5-5 pour les 600 instances J120.

Tableau 5-3 : Résultats de la résolution des instances J30 du RCPSP avec les trois procédés de discrétisation.

	Priorité d'ordonnancement			Liste de translations			Liste de permutations		
Exéc.	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé
1	0,51%	6,90%	79,79%	0,32%	5,88%	86,25%	0,16%	3,45%	91,04%
2	0,55%	6,12%	79,58%	0,28%	5,15%	86,46%	0,15%	3,45%	90,83%
3	0,53%	6,78%	80,21%	0,34%	7,14%	85,63%	0,21%	6,19%	89,17%
4	0,54%	6,19%	78,96%	0,16%	6,17%	86,04%	0,17%	4,35%	91,25%
5	0,51%	6,90%	79,79%	0,29%	7,41%	86,67%	0,16%	3,45%	90,83%
Moy.	**0,53%**	**6,58%**	**79,67%**	**0,28%**	**6,35%**	**86,21%**	**0,17%**	**4,18%**	**90,63%**

Tableau 5-4 : Résultats de la résolution des instances J60 du RCPSP avec les trois procédés de discrétisation.

	Priorité d'ordonnancement			Liste de translations			Liste de permutations		
Exéc.	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé
1	12,70%	116,88%	59,17%	12,00%	111,69%	60,42%	11,86%	111,69%	60,42%
2	12,73%	115,58%	59,38%	12,03%	110,39%	60,00%	11,92%	109,09%	59,79%
3	12,65%	114,29%	59,17%	12,05%	111,69%	59,79%	11,90%	111,69%	60,42%
4	12,72%	115,58%	58,96%	11,86%	110,39%	59,38%	11,86%	112,99%	60,42%
5	12,71%	120,78%	58,96%	12,03%	111,69%	60,21%	11,91%	115,58%	60,21%
Moy.	**12,70%**	**116,62%**	**59,13%**	**11,99%**	**111,17%**	**59,96%**	**11,89%**	**112,21%**	**60,25%**

Tableau 5-5 : Résultats de la résolution des instances J120 du RCPSP avec les trois procédés de discrétisation.

	Priorité d'ordonnancement			Liste de translations			Liste de permutations		
Exéc.	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé	Moy. dév.	Dév. Max.	% trouvé
1	36,78%	218,18%	21,83%	35,98%	211,11%	21,50%	36,66%	217,17%	22,33%
2	36,83%	216,16%	22,33%	35,97%	210,10%	22,17%	36,58%	216,16%	22,17%
3	36,81%	220,20%	22,00%	35,99%	209,09%	21,50%	36,61%	213,13%	21,67%
4	36,75%	217,17%	21,83%	35,93%	211,11%	21,83%	36,66%	217,17%	22,00%
5	36,75%	214,14%	21,17%	36,06%	207,92%	22,50%	36,62%	212,87%	21,67%
Moy.	**36,78%**	**217,17%**	**21,83%**	**35,99%**	**209,87%**	**21,90%**	**36,62%**	**215,30%**	**21,97%**

En examinant les tableaux précédents, il est possible de remarquer que le procédé de discrétisation avec liste de permutations est celui proposant les plus petites moyennes de déviation aux solutions optimales pour les instances J30 avec 0,17% et aux meilleures solutions connues pour les instances J60 avec 11,89%. Il s'agit également du procédé ayant trouvé le plus grand nombre de solutions avec 90,63% et 60,25% pour ces mêmes groupes

d'instances, proposant ainsi les meilleures performances en comparaison avec les autres procédés. Le procédé de discrétisation par liste de translations est deuxième et celui par priorité d'ordonnancement est troisième pour ces mêmes instances.

En revanche, les résultats deviennent moins clairs avec une augmentation de la taille des instances. En effet, le procédé de discrétisation par liste de translations est celui proposant les plus petites moyennes de déviations et de déviations maximales en comparaison avec les meilleures solutions connues sur les instances J120, suivi de très près par les deux autres procédés. La différence est cependant peu significative. La force principale de l'utilisation des listes de translations réside dans l'obtention d'une plus petite moyenne de déviation maximale en comparaison avec les deux autres procédés de discrétisation.

En ce qui concerne l'autre point de comparaison, la rapidité de calculs, le Tableau 5-6 fait état des temps d'exécution en secondes pour les trois procédés de discrétisation, sur les trois ensembles d'instances du RCPSP (J30, J60 et J120).

Tableau 5-6 : Temps d'exécution en secondes des trois procédés de discrétisation en fonction de la taille des instances du RCPSP.

	Priorité d'ordonnancement			Liste de translations			Liste de permutations		
Exéc.	**J30**	**J60**	**J120**	**J30**	**J60**	**J120**	**J30**	**J60**	**J120**
1	9,52	61,68	486,92	9,90	65,18	506,27	9,52	62,54	484,89
2	9,64	61,76	486,89	9,90	65,05	507,92	9,51	62,57	485,20
3	9,56	61,76	487,49	9,89	65,22	507,21	9,53	62,67	485,50
4	9,51	61,81	487,03	9,88	64,95	506,65	9,52	62,43	485,01
5	9,52	61,55	487,07	9,85	65,09	506,18	9,55	62,71	485,51
Moy.	9,55	61,71	487,08	9,88	65,10	506,85	9,53	62,58	485,22

À la lumière de ces résultats, il est possible d'affirmer que le procédé de discrétisation le plus rapide est celui utilisant les priorités d'ordonnancement, et ce, pour les trois catégories d'instances du RCPSP. Le procédé le plus lent est celui par liste de translations. Toutefois, la différence en termes de temps d'exécution est moins significative pour le

RCPSP que pour le SOP. La difficulté de distinguer les temps d'exécution des trois procédés de discrétisation lors de la résolution du RCPSP provient de la définition du problème. En effet, la nature fortement contrainte du RCPSP implique une gestion de ressources, laquelle engendre potentiellement des temps non négligeables consacrés à la réparation des solutions. Ces temps de réparation tendent à aplanir les temps d'exécution. De ce fait, il devient difficile de distinguer efficacement la rapidité de la mise à jour des particules pour les différents procédés de discrétisation et le temps utilisé pour rendre les solutions valides.

5.4.3. ANALYSE COMPARATIVE DES RÉSULTATS

La performance obtenue par les trois procédés de discrétisation sur les deux problèmes de support permet d'arriver à certaines conclusions. La résolution du SOP suggère une préférence pour la liste de translations considérant les plus petites déviations aux meilleures solutions connues selon les instances testées. La résolution du RCPSP suggère quant à elle que le procédé de discrétisation par liste de permutations offre de meilleurs résultats pour la résolution des petites instances (J30), et des résultats similaires que ceux du procédé par liste de translations pour les instances moyennes (J60). Cependant, le procédé par liste de translations offre des performances légèrement meilleures sur la résolution de grandes instances de problèmes (J120), mais ne permet pas de définir une dominance comparativement aux deux autres procédés de discrétisation.

La difficulté de différencier la performance des procédés de discrétisation pour la résolution du RCPSP peut être explicable par la nature du problème. En effet, le RCPSP est plus contraint que le SOP étant donné la nécessité de respecter à la fois les contraintes de ressources et celles des préséances, alors que le SOP n'a que les contraintes de préséances à satisfaire. La présence d'un grand nombre de contraintes entraîne un aplanissement des solutions en raison de l'obligation d'effectuer une réparation des solutions obtenues après une mise à jour des particules. En ce sens, un problème fortement contraint

permettrait plus difficilement de départager la performance des méthodes de discrétisation. À l'opposé, un problème plus faiblement contraint comme le SOP permet de conclure à une meilleure stratégie de conception par l'utilisation d'un procédé de discrétisation par liste de translations. Une corrélation peut donc possiblement être observée entre le niveau de contraintes d'un problème et la capacité à distinguer l'efficacité d'un procédé de discrétisation.

Au sujet du temps d'exécution, le procédé de discrétisation par priorité d'ordonnancement est plus rapide que les deux autres procédés lors de la résolution du RCPSP. Il s'agit en contrepartie du procédé de discrétisation le plus lent pour la résolution du SOP. Plusieurs hypothèses peuvent être émises par de tels résultats : la nature du problème et la dimension de la vélocité d'une particule.

Tout d'abord, la nature fortement contrainte du RCPSP implique qu'un temps d'exécution important est utilisé pour la réparation des solutions, laquelle est un processus commun et coûteux pour chacun des procédés de discrétisation. La différence entre les procédés pour résoudre les instances du SOP est plus facile à observer étant donné que ce problème présente un plus faible niveau de contraintes. En conséquence, il utilise moins de temps d'exécution lors de la réparation des préséances et une plus grande majorité du temps est consacrée à la mise à jour des particules, de même qu'à l'évaluation des solutions par la fonction objectif.

5.5. CONCLUSION SUR LA CONTRIBUTION

Ce chapitre porte sur l'utilisation de procédés de discrétisation lors de la résolution de problèmes par les métaheuristiques. Il débute avec la présentation de deux problèmes combinatoires servant de support aux expérimentations effectuées, soit le SOP et le RCPSP. Trois procédés de discrétisation provenant de la Section 3.4 ont ensuite été sélectionnés et

présentés : par priorité d'ordonnancement (H. Zhang et al., 2006), par liste de translations (Anghinolfi & Paolucci, 2009) et par liste de permutations (Clerc, 2004).

Après avoir été définis, les trois procédés de discrétisation sont utilisés pour résoudre les deux problèmes de support, par comparaison avec la moyenne des déviations, les déviations maximales et le temps de résolution sur une grande variabilité d'instances. Le but de cette comparaison était de valider l'apport de ces procédés de discrétisation lors de la résolution de problèmes combinatoires en les comparant avec une base commune. Il s'agissait donc de comparer les comportements et les avantages, de même que les inconvénients.

À la suite des tests, il est possible de déterminer que l'augmentation de la dimension d'un problème et l'utilisation d'un problème comportant un grand nombre de contraintes tend à rendre difficile la comparaison entre différents procédés de discrétisation. Cependant, le procédé de discrétisation proposant les meilleurs avantages est celui par liste de translations. Pour arriver à cette conclusion, il a été possible d'observer qu'il s'agit du procédé comportant le plus bas pourcentage de valeurs déviant de la meilleure solution connue lors de l'utilisation de grandes instances de SOP et de RCPSP. Bien que la différence ne fut pas la plus importante pour la résolution du RCPSP, elle était plus significative lors de la résolution du SOP. Cependant, son temps de résolution était intermédiaire entre les trois procédés de discrétisation. Il ne s'agissait pas du procédé le plus rapide, ni celui le plus lent. Son temps de résolution demeure tout de même négligeable considérant le gain obtenu par la qualité des solutions lors de l'utilisation du PSO pour la résolution de problèmes combinatoires.

Plus globalement, ce chapitre permet de mieux comprendre et d'améliorer la résolution de problèmes comportant des variables de décision entières avec l'utilisation du PSO. Il a ainsi permis d'*Adapter et de comparer trois méthodes de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support*. Parmi la

grande variété des procédés de discrétisation provenant de la littérature, le procédé par liste de translations d'Anghinolfi et Paolucci (2009) semble le plus prometteur. Il est le procédé de discrétisation ayant permis d'offrir au PSO une performance moyenne robuste lors de la résolution de problèmes combinatoires à complexité variable, et ce, avec des instances de différentes tailles.

CHAPITRE 6 - CONCEPTION D'UN PSO HYBRIDE ET DISCRET POUR LA RÉSOLUTION DE GRANDES INSTANCES DU PROBLÈME D'ALIGNEMENT MULTIPLE DE SÉQUENCES

Cette section fait l'objet d'actes de conférence soumis :

> Deschênes, H., Gagné, C. (2022). *Solving the MSA using a New Ring Topology with a Discrete Hybrid PSO*.
>
> Deschênes, H., Gagné, C. (2022). *Using Magnetism with PSO Hybrid Variants to Solve the Multiple Sequence Alignment Problem*.

Une version étendue de ces références est également en préparation pour une revue scientifique.

6.1. INTRODUCTION

Ce chapitre représente le point culminant de cette thèse par la combinaison des conclusions provenant des expérimentations du Chapitre 4 et du Chapitre 5. Au Chapitre 4, il est déterminé que l'hybridation offrant les performances les plus prometteuses est le HCoCLPSO, avec la combinaison du processus d'apprentissage compréhensif du CLPSO et de la structure comportementale du CoLPSO. Le Chapitre 5 a permis de déterminer que la discrétisation offrant les performances moyennes les plus robustes est celle utilisant une liste de translations d'Anghinolfi et Paolucci (2009). Ce procédé de discrétisation et cette d'hybridation doivent être adaptés adéquatement au problème à résoudre afin de confirmer leur performance. Avec l'utilisation de ces stratégies, deux nouveaux comportements sont proposés pour la résolution d'un problème complexe de grande taille : l'alignement multiple de séquences (MSA).

Tout d'abord, la Section 6.2 développe la définition du problème de support utilisé : le MSA. Les enjeux, la fonction objectif et les différentes méthodes de résolution reconnues dans le domaine sont décrits.

La première contribution est par la suite présentée à la Section 6.3 : la nouvelle discrétisation du PSO avec une modélisation de solutions en anneau. Pour établir la discrétisation la plus performante pour résoudre le MSA, le procédé par liste de translations d'Anghinolfi et Paolucci (2009) est adapté et comparé de deux manières différentes : par l'utilisation de variables de décision entières et réelles. Pour ce faire, une modélisation de solutions sous forme d'anneau et spécialisée pour le MSA est proposée dans le cadre de cette discrétisation.

Une fois cette expérimentation complétée, la deuxième contribution est présentée à la Section 6.4 : le processus d'apprentissage par magnétisme. Ce nouveau processus permet

de concevoir une nouvelle hybridation basée sur le HCoCLPSO pour résoudre le MSA. Cette contribution consiste à remplacer le processus d'apprentissage compréhensif du CLPSO par un nouveau processus d'apprentissage mieux adapté au problème à résoudre. Ce nouveau processus crée une attraction entre les différents espacements à l'intérieur d'une séquence pour les aider à se regrouper et ainsi diminuer les pénalités entraînées dans l'évaluation par la fonction objectif. La deuxième portion des expérimentations de ce chapitre consiste à attribuer adéquatement une valeur pour le paramètre définissant le nombre maximal d'appels au magnétisme pour la résolution du MSA. Deux techniques d'attribution sont utilisées : l'attribution statique et l'attribution dynamique. Une fois ce paramètre défini adéquatement, les résultats sont comparés avec la contribution de la Section 6.3, de même qu'avec deux outils commerciaux utilisés en bio-informatique : MAFFT et CLUSTAL.

6.2. PRÉSENTATION DES PROBLÈMES D'ALIGNEMENT DE DE SÉQUENCES

6.2.1. CONTEXTE BIOLOGIQUE

La théorie de l'évolution des espèces selon Darwin indique que tout organisme vivant provient d'un ancêtre commun et évolue afin de s'adapter à son environnement. Par exemple, l'être humain est un organisme vivant considéré plus près du chimpanzé dans l'évolution des espèces que de la baleine bleue grâce aux processus évolutionnaires. Le degré de similarité entre deux espèces peut être mesuré par la comparaison de leur matériel génétique. Il permet d'identifier la proximité entre ces espèces dans l'arbre de vie.

L'évolution des espèces se résume à trois facteurs affectant la composition de l'acide désoxyribonucléique (ADN) d'une espèce : la suppression, l'ajout et la mutation d'un caractère de l'ADN (Pevsner, 2015). Un seul changement peut affecter la lecture de la chaîne ADN et, par le fait même, l'expression d'une protéine. L'ADN contient toute

l'information génétique d'un individu. Il est composé de nucléotides représentés par quatre lettres de l'alphabet : A, C, T et G. Après un processus de traduction et de transfert à l'intérieur du corps humain, différentes combinaisons de nucléotides permettent de produire des acides aminés. Ces derniers sont représentés avec une équivalence au système de notation de l'ADN par l'utilisation de vingt lettres de l'alphabet au lieu de quatre : A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W et Y. L'ensemble d'une série d'acides aminés compose une protéine, permettant la production de gènes chez un individu. L'agencement des protéines à l'intérieur du matériel génétique d'un être vivant génère des caractéristiques uniques telles que la grandeur, un trait de caractère, une maladie ou encore la composition du fonctionnement des organes internes d'un mammifère (Pevsner, 2015). La comparaison et l'alignement entre plusieurs chaînes d'ADN ou d'acides aminés correspondent au problème d'alignement multiple de séquences.

Dans le domaine de la bio-informatique, le MSA peut être utile afin de rechercher l'homologie entre différents gènes d'espèces différentes, pour annoter les génomes, pour modéliser des sites de liaisons ou encore pour la construction d'arbres phylogénétiques (Bawono et al., 2017). Ces arbres sont des représentations visuelles de l'évolution des espèces selon un gène donné. Le taux de ressemblance entre plusieurs séquences pour un gène provenant de différentes espèces permet d'observer leur proximité dans l'évolution. Le MSA est également utilisé dans une autre discipline, dont en littérature, afin de reconstruire l'historique de l'origine des mots en se basant sur la correspondance phonétique en différentes langues (Jäger, 2015; Prokić et al., 2009), de même que pour détecter des paraphrases à l'intérieur d'un texte par la reconnaissance de modèles existants (Barzilay & Lee, 2003).

6.2.2. DÉFINITION DES PROBLÈMES D'ALIGNEMENT DE SÉQUENCES

La préférence de la résolution de l'alignement de séquences d'acides aminés plutôt que de séquences ADN s'explique par la qualité de l'information obtenue (Pevsner, 2015). En effet, plusieurs modifications dans une séquence d'ADN ne changeront pas nécessairement l'acide aminé qui en résulte. Sachant que plusieurs acides aminés partagent des propriétés similaires, la relation observée entre eux permet l'utilisation d'un système de notation sous la forme d'une matrice de substitution. Le niveau de ressemblance entre chaque acide aminé peut alors être évalué, tout comme leur probabilité de mutation. La comparaison entre les séquences d'acides aminés permet l'identification de séquences homologues, ce que la comparaison des séquences ADN correspondantes ne permet pas (Pearson, 1996). Le terme « homologue » est utilisé ici afin de définir des segments ayant un ancêtre commun, et dont une des séquences dérive d'une autre dans l'évolution des espèces. Deux séquences sont donc homologues à un certain degré lorsqu'elles ont un ancêtre commun. Une forte ressemblance suggère également que les séquences alignées occupent des fonctions similaires.

Le MSA est un problème de nature combinatoire dit NP-difficile (Lusheng Wang & Jiang, 1994) utilisé principalement dans le domaine de la bio-informatique. Il s'agit d'un problème composé de variables de décision de nature entière représentant soit des séquences à aligner, soit l'emplacement des espaces générés au sein de l'alignement multiple de séquences d'acides aminés. Le but est de produire un arrangement de plus de deux séquences en les superposant, puis en alignant les caractères les composant afin d'obtenir la plus forte ressemblance possible. Il y a recherche du plus haut taux de similarité selon la fonction objectif. Pour y parvenir, le rôle de l'algorithme est d'insérer des espaces à l'intérieur des séquences à aligner dans le but d'augmenter le degré de similarité entre elles. Afin d'étudier le problème d'alignement multiple de séquences, il est nécessaire de voir tout d'abord les principes de l'alignement simple de séquences.

6.2.2.1. ALIGNEMENT SIMPLE DE SÉQUENCES

Le but de la résolution d'un alignement simple est de maximiser la similarité entre deux séquences par l'alignement vertical de caractères lorsque les deux séquences sont mises une au-dessus de l'autre. Pour ce faire, des espaces sont générés au sein des séquences afin de simuler la suppression, l'ajout et la mutation d'un caractère de l'ADN. Ceci permet de créer un alignement global comportant des séquences de longueurs identiques, considérant que les séquences de base ne sont pas de la même longueur. Chaque paire de caractères est alignée et évaluée selon une matrice de substitution. La somme obtenue par l'évaluation des paires de caractères est diminuée par des pénalités applicables selon les espacements générés lors de l'alignement. En effet, la résolution de ce problème dépend du nombre d'*espacements* dans les séquences (le nombre d'*ouvertures*), de même que la longueur de ceux-ci (les *extensions*) (Katoh & Standley, 2016).

Les séquences utilisées lors de la résolution de l'alignement de séquences n'ont pas l'obligation d'être de la même longueur. Cette longueur constitue un paramètre qu'il est possible de définir en début d'algorithme. Certains auteurs proposent que cette longueur doit être celle de la plus longue séquence en permettant un rallongement de 20% à 50% (Chellapilla & Fogel, 1999). D'autres auteurs proposent également une longueur maximale excédentaire de 20% de la plus longue séquence utilisée (Lalwani et al., 2013; Moustafa et al., 2017).

La Figure 6-1 illustre un exemple du résultat d'un alignement simple entre deux sous-séquences d'acides aminés, nommées *Séq. 1* et *Séq. 2*. La ligne du centre située entre les sous-séquences *Séq. 1* et *Séq. 2* indique les caractères identiques par une lettre. Avec cet exemple, 31,1% (14/45) du contenu des deux sous-séquences est identique après l'alignement. Chaque tiret au sein d'une sous-séquence indique la présence d'un espace ayant été généré lors de l'alignement, et représente donc l'absence de caractère. Au total,

dix espaces sont ajoutés à *Séq. 1* pour une longueur d'alignement total de 45 caractères. Trois espaces sont également ajoutés à *Séq. 2* pour obtenir la même longueur.

```
Séq. 1 : ------MQDRVKRPMN---VFFRDQRRKMA-LNPRMRNSEISKIL
                 D  K P      FF   R K A   P M N    KIL
Séq. 2 : MKKLKKHPDFPKKPLTPYFRFF---RAKYAKLHPEMSNLDLTKIL
```

Figure 6-1 : Exemple d'alignement simple de deux séquences d'acides aminés.
© Hugo Deschênes

6.2.2.2. MATRICES DE SUBSTITUTION

Plusieurs matrices de substitution existent afin d'évaluer un alignement. Le contenu d'une telle matrice est défini par une valeur proportionnelle de la probabilité qu'un acide aminé mute en un autre. Elles sont construites en analysant l'alignement d'une grande quantité de séquences d'acides aminés prouvées homologues. Le contenu reflète donc la probabilité de mutation de chaque acide aminé au cours de l'évolution des espèces. Les deux modèles de matrice de substitution les plus utilisés sont PAM (M. O. Dayhoff, 1972) et BLOSUM (Henikoff & Henikoff, 1992).

La spécificité de la matrice PAM est qu'elle utilise, pour la définition des valeurs qui la compose, des alignements de séquences provenant d'espèces très près dans l'évolution (avec seulement 1% de différence entre les pairs d'acides aminés). La matrice résultante, PAM1, est donc définie selon les ressemblances obtenues et les probabilités pour un acide aminé de ressembler à un autre. Elle a servi de base pour l'élaboration de la matrice qui est maintenant utilisée aujourd'hui, PAM250. Cette dernière est obtenue en multipliant la matrice originale (PAM1) 250 fois avec elle-même. Elle est utilisée dans les cas où il y a un maximum de 20% de similarité entre les séquences, en supposant que cette information est connue avant la résolution de l'alignement. La Figure 6-2 illustre le contenu de la matrice de substitution PAM250.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	J	Z	X	*
A	2	-2	0	0	-2	0	0	1	-1	-1	-2	-1	-1	-3	1	1	1	-6	-3	0	0	-1	0	-1	-8
R	-2	6	0	-1	-4	1	-1	-3	2	-2	-3	3	0	-4	0	0	-1	2	-4	-2	-1	-3	0	-1	-8
N	0	0	2	2	-4	1	1	0	2	-2	-3	1	-2	-3	0	1	0	-4	-2	-2	2	-3	1	-1	-8
D	0	-1	2	4	-5	2	3	1	1	-2	-4	0	-3	-6	-1	0	0	-7	-4	-2	3	-3	3	-1	-8
C	-2	-4	-4	-5	12	-5	-5	-3	-3	-2	-6	-5	-5	-4	-3	0	-2	-8	0	-2	-4	-5	-5	-1	-8
Q	0	1	1	2	-5	4	2	-1	3	-2	-2	1	-1	-5	0	-1	-1	-5	-4	-2	1	-2	3	-1	-8
E	0	-1	1	3	-5	2	4	0	1	-2	-3	0	-2	-5	-1	0	0	-7	-4	-2	3	-3	3	-1	-8
G	1	-3	0	1	-3	-1	0	5	-2	-3	-4	-2	-3	-5	0	1	0	-7	-5	-1	0	-4	0	-1	-8
H	-1	2	2	1	-3	3	1	-2	6	-2	-2	0	-2	-2	0	-1	-1	-3	0	-2	1	-2	2	-1	-8
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5	2	-2	2	1	-2	-1	0	-5	-1	4	-2	3	-2	-1	-8
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6	-3	4	2	-3	-3	-2	-2	-1	2	-3	5	-3	-1	-8
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5	0	-5	-1	0	0	-3	-4	-2	1	-3	0	-1	-8
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6	0	-2	-2	-1	-4	-2	2	-2	3	-2	-1	-8
F	-3	-4	-3	-6	-4	-5	-5	-5	-2	1	2	-5	0	9	-5	-3	-3	0	7	-1	-4	2	-5	-1	-8
P	1	0	0	-1	-3	0	-1	0	0	-2	-3	-1	-2	-5	6	1	0	-6	-5	-1	-1	-2	0	-1	-8
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	2	1	-2	-3	-1	0	-2	0	-1	-8
T	1	-1	0	0	-2	-1	0	0	-1	0	-2	0	-1	-3	0	1	3	-5	-3	0	0	-1	-1	-1	-8
W	-6	2	-4	-7	-8	-5	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	17	0	-6	-5	-3	-6	-1	-8
Y	-3	-4	-2	-4	0	-4	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	0	10	-2	-3	-1	-4	-1	-8
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	-2	4	-2	2	-2	-1	-8
B	0	-1	2	3	-4	1	3	0	1	-2	-3	1	-2	-4	-1	0	0	-5	-3	-2	3	-3	2	-1	-8
J	-1	-3	-3	-3	-5	-2	-3	-4	-2	3	5	-3	3	2	-2	-2	-1	-3	-1	2	-3	5	-2	-1	-8
Z	0	0	1	3	-5	3	3	0	2	-2	-3	0	-2	-5	0	0	-1	-6	-4	-2	2	-2	3	-1	-8
X	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-8
*	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	1

Figure 6-2 : Matrice de substitution PAM250 utilisée pour l'alignement de séquences d'acides aminés (M. Dayhoff et al., 1978).

Une alternative à PAM250 est l'utilisation de la matrice BLOSUM (*BLOck SUbstitution Matrix*). Cette dernière a été développée avec l'utilisation d'une base de données comportant près de 500 groupes d'alignements par pairs. Elle est construite sur des alignements observés et non sur des extrapolations provenant de la comparaison d'acides aminés homologues (comme c'est le cas de PAM) (Henikoff & Henikoff, 1992). BLOSUM62, la version la plus utilisée de cette matrice, est surtout performante pour un alignement d'acides aminés dont les séquences possèdent au moins 62% de similitude. La Figure 6-3 illustre le contenu de la matrice BLOSUM62.

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	−1	5																		
Asn	−2	0	6																	
Asp	−2	−2	1	6																
Cys	0	−3	−3	−3	9															
Gln	−1	1	0	0	−3	5														
Glu	−1	0	0	2	−4	2	5													
Gly	0	−2	0	−1	−3	−2	−2	6												
His	−2	0	1	−1	−3	0	0	−2	8											
Ile	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

Figure 6-3 : Matrice de substitution BLOSUM62 utilisée pour l'alignement de séquences d'acides aminés (Henikoff & Henikoff, 1992).
Figure en libre accès

Lors d'un alignement, le choix de la matrice de substitution doit se faire en fonction du degré de ressemblance soupçonné entre les deux séquences. Un haut numéro de PAM et un bas numéro de BLOSUM tendent à correspondre à une matrice comportant des valeurs pour des séquences plus divergentes. Un exemple de divergence est observable par la comparaison du génome humain avec celui d'une bactérie (Pevsner, 2015). La Figure 6-4 illustre un résumé du choix de la matrice de substitution à effectuer entre BLOSUM et PAM en fonction de la divergence de l'alignement.

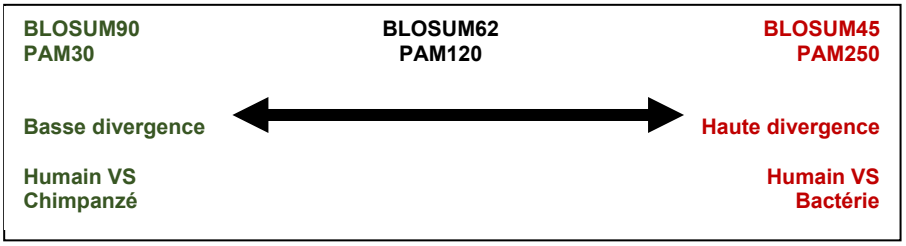

Figure 6-4 : Principales utilisations des matrices de substitution BLOSUM et PAM.
© Figure adaptée avec l'autorisation de Wiley

6.2.2.3. ALIGNEMENT MULTIPLE DE SÉQUENCES

Tel que son nom l'indique, le MSA consiste à effectuer un alignement de plus de deux séquences. Le but de ce problème consiste à aligner les caractères similaires afin de produire un résultat permettant d'obtenir le score le plus élevé possible avec l'évaluation par la fonction objectif. La Figure 6-5 illustre un exemple d'alignement multiple de séquences d'acides aminés entre quatre séquences provenant de l'ensemble de données *BaliBase3.0* (Bahr et al., 2001; Thompson et al., 2005).

```
1aab_  : ---GKGDPKKPRGKMSSYAFFVQTSREEHKKKHPDASVNFSEFSKKCSERWKTMSAKEKGKFEDMAKADKARYEREMKTYIPPKGE----------
1j46_A : ------MQDRVKRPMNAFIVWSRDQRRKMALENP--RMRNSEISKQLGYQWKMLTEAEKWPFFQEAQKLQAMHREKYPNYKYRPRRKAKMLPK---
1k99_A : MKKLKKHPDFPKKPLTPYFRFFMEKRAKYAKLHP--EMSNLDLTKILSKKYKELPEKKKMKYIQDFQREKQEFERNLARFREDHPDLIQNAKK---
2lef_A : --------MHIKKPLNAFMLYMKEMRANVVAEST--LKESAAINQILGRRWHALSREEQAKYYELARKERQLHMQLYPGWSARDNYGKKKKRKREK
```

Figure 6-5 : Exemple d'alignement multiple de séquences d'acides aminés provenant de l'instance BB11001 de *BaliBase3.0* (Bahr et al., 2001; Thompson et al., 2005)
© Hugo Deschênes

Pour évaluer la qualité d'un alignement multiple de séquences d'acides aminés, la fonction objectif calcule le pointage $\boldsymbol{\alpha}$ associé à chaque paire de caractères alignés selon la matrice de substitution utilisée, et ce, pour toutes les paires de séquences de l'alignement global. Le pointage $\boldsymbol{\beta}$ est ensuite calculé, lequel représente la somme des pénalités reliées aux espaces générés. Afin de calculer $\boldsymbol{\beta}$, il est nécessaire de définir une valeur $\boldsymbol{\varepsilon}$ pour une *pénalité d'ouverture d'un espacement* au sein d'une séquence ainsi qu'une valeur $\boldsymbol{\delta}$ pour une *pénalité d'agrandissement de cet espacement*. Les valeurs de $\boldsymbol{\varepsilon}$ et $\boldsymbol{\delta}$ correspondent à deux paramètres à définir avant le début d'un alignement (Pevsner, 2015). J. Sun et al. (2014) définissent la somme de toutes les pénalités $\boldsymbol{\beta}$ d'un alignement (Équation 6-2), laquelle est soustraite au calcul du pointage de chaque paire de séquences $\boldsymbol{\alpha}$ (Équation 6-1). Ceci permet d'obtenir la valeur finale de la fonction objectif, nommée *SP* dans la littérature et définie par l'Équation 6-3. Dans ces équations, s_j et s_k représentent respectivement les séquences alignées j et k, $distance$ correspond au calcul du pointage entre les séquences, S

est le nombre total de séquences et z est la longueur d'un espacement e pour tous les E espacements.

$$\alpha = \sum_{j=1}^{S-1} \sum_{k=j+1}^{S} distance(s_j, s_k) \quad (6\text{-}1)$$

$$\beta = \varepsilon + (z-1) \cdot \delta \quad (6\text{-}2)$$

$$SP = \alpha - \sum_{e=1}^{E} \beta_e \quad (6\text{-}3)$$

6.2.3. DESCRIPTION DES INSTANCES TESTS

Afin de comparer la performance des algorithmes lors de la résolution du MSA, Thompson et al. (2005) proposent plusieurs catégories d'instances de référence regroupées selon certaines caractéristiques précises. Ces instances sont basées sur le travail effectué par Bahr et al. (2001). Cette banque d'instances se nomme BAliBASE 3.0. Par exemple, les instances provenant de la référence RV20 contiennent toutes une séquence dite « orpheline », c'est-à-dire qu'elle ne possède aucun lien ni ressemblance avec les autres séquences provenant de la même instance. Ce chapitre traite des instances provenant de la référence RV11. Celle-ci contient 38 instances dont les séquences possèdent moins de 20% de caractères similaires. Le Tableau 6-1 décrit ces instances et identifie le nombre de séquences (Nb. Séq.), de même que la longueur de la plus petite (+Court) et de la plus longue (+Long) séquence à aligner. La dimension (Dim.) y est également identifiée, laquelle représente le nombre d'espaces à ajouter à l'alignement multiple pour que la longueur de chaque séquence corresponde à la plus longue séquence, additionnée du 20% supplémentaire. La valeur de la différence entre la longueur de la plus longue séquence et de la plus courte (Diff. Long.) est représentée dans la dernière colonne.

Tableau 6-1 : Détails des instances provenant des références RV11 de BAliBASE 3.0 (Thompson et al., 2005).

Instances	Dim.	Nb. Séq.	+ Court	+ Long	Diff. Long.
BB11001	91	4	83	91	8
BB11002	1064	8	52	193	141
BB11003	596	4	414	516	102
BB11004	504	4	390	456	66
BB11005	2230	14	329	465	136
BB11006	754	8	186	283	97
BB11007	1167	9	385	457	72
BB11008	1372	4	104	540	436
BB11009	752	4	97	337	240
BB11010	394	4	490	492	2
BB11011	445	5	160	242	82
BB11012	448	4	320	397	77
BB11013	247	5	51	101	50
BB11014	1323	6	502	634	132
BB11015	334	4	297	327	30
BB11016	3228	8	316	729	413
BB11017	253	4	247	264	17
BB11018	4929	14	418	750	332
BB11019	1266	10	299	396	97
BB11020	583	9	201	237	36
BB11021	195	4	102	139	37
BB11022	512	4	63	205	142
BB11023	1247	7	231	407	176
BB11024	604	4	372	465	93
BB11025	161	4	64	103	39
BB11026	5929	7	76	906	830
BB11027	1998	7	175	432	257
BB11028	942	10	93	211	118
BB11029	252	4	81	138	57
BB11030	2495	14	236	392	156
BB11031	3924	11	300	611	311
BB11032	1506	8	226	403	177
BB11033	1482	11	85	239	154
BB11034	3059	8	401	729	328
BB11035	338	5	71	138	67
BB11036	1224	8	298	436	138
BB11037	4533	5	335	1192	857
BB11038	2974	8	261	614	353

Dans le cadre de ce chapitre, les instances non grisées du Tableau 6-1 sont sélectionnées pour valider les expérimentations. Cette limite est établie afin de permettre la résolution d'un grand nombre d'instances dans des délais raisonnables, considérant que le

calcul haute performance n'est pas utilisé et que les instances sont très grandes. Cette limite correspond à toutes les instances dont la dimension est inférieure à 1500. Les 27 instances correspondant à ce critère sont variées et permettent d'effectuer une analyse de la performance des contributions de ce chapitre.

6.2.4. PRINCIPALES APPROCHES DE RÉSOLUTION DU MSA RECONNUES DANS LE DOMAINE

Selon la nomenclature de Notredame (2004), trois grandes approches sont recensées pour effectuer l'alignement multiple de séquences : *exacte*, *progressive* et *itérative*. D'autres approches sont mentionnées dans son ouvrage et correspondent à des dérivées de ces trois dernières. Bien qu'elle permette l'obtention de solutions optimales, l'approche *exacte* n'est viable que pour un faible nombre de séquences. Le temps d'exécution augmente considérablement avec l'ajout de séquences à l'alignement multiple. L'approche *progressive* ajoute une séquence à la fois à l'alignement global, en prenant bien soin de ne jamais aligner plus de deux séquences en même temps. Elle utilise la programmation dynamique. Un outil populaire de l'approche *progressive* est *CLUSTALW* (ou sa variante *CLUSTAL X*) (Hung et al., 2015; Larkin et al., 2007). L'intérêt de cet outil est surtout sa simplicité et sa rapidité. Il n'est cependant pas en mesure de garantir un niveau d'optimisation précis et un seuil minimum de qualité des solutions (Pervez et al., 2014). Finalement, l'approche *itérative* se base sur un algorithme existant capable de produire un alignement, reprenant ensuite le résultat et en le peaufinant de manière itérative jusqu'à ce qu'aucune amélioration ne soit produite. *MAFFT* (Katoh et al., 2002) et *MUSCLE* (Edgar, 2004a, 2004b) sont deux exemples d'outils utilisant l'approche *itérative*.

En plus de ces trois grandes approches, Pevsner (2015) en mentionne deux autres aussi importantes : une approche *basée sur la cohérence* et une approche *basée sur la structure*. L'approche *basée sur la cohérence* combine les approches *itérative* et *progressive*.

Elle se base sur toutes les possibilités d'alignement entre paires de séquences en optimisant chacun des caractères afin de produire un alignement global le plus fidèle possible. *T-COFFEE* (Notredame et al., 2000) et *ProbCons* (Do et al., 2005) sont deux exemples d'outils dans cette catégorie. L'autre approche mentionnée, celle *basée sur la structure*, consiste à prendre en considération la structure du repliement des protéines produites par les acides aminés. Une structure similaire entre différentes protéines a plus tendance à correspondre à une séquence similaire d'acides aminés. Un exemple d'outil utilisant cette approche est le module *Expresso* de *T-COFFEE* (Armougom et al., 2006).

Malgré la variabilité des outils, il est impossible d'affirmer qu'un d'entre eux performe mieux que les autres. Chaque outil a ses avantages sur différents alignements de séquences selon la ressemblance, la longueur et la quantité de séquences à aligner (Kumar, 2015; Pais et al., 2014; Pervez et al., 2014). Bien que plusieurs d'entre eux soient accessibles, la capacité à obtenir des résultats d'une grande fiabilité peut être améliorée (Bawono et al., 2017).

Depuis quelques années, beaucoup d'avancements sont faits pour la résolution du MSA par l'utilisation de métaheuristiques. Le recuit simulé (Kim et al., 1994), la recherche du coucou (Kartous et al., 2014) de même que la colonie d'abeilles artificielles (Öztürk & Aslan, 2016; Rubio-Largo et al., 2016) font partie des métaheuristiques ayant servi à la résolution de ce problème. Plus de travaux ont cependant été réalisés avec l'algorithme génétique (Botta & Negro, 2010; Kaya et al., 2014), ou bien avec une version hybride incluant cette métaheuristique (Lee et al., 2008; Naznin et al., 2011). Selon ces derniers auteurs, le succès de l'algorithme génétique est observé principalement lors de la résolution de petites instances de problèmes, donc par l'utilisation de petites séquences d'acides aminés. Des auteurs ont préféré avoir recours au PSO afin de pallier cette limitation. Avec cette métaheuristique, il est possible d'utiliser moins d'espace mémoire pour résoudre le MSA,

permettant alors l'optimisation d'un plus grand nombre de séquences et de séquences plus longues (Lalwani et al., 2015; Mohamed & Aboul Ella, 2017).

En effet, beaucoup de travaux ont été publiés depuis quelques années en utilisant le PSO comme choix de métaheuristique pour la résolution du MSA (Kamal et al., 2012; Lalwani et al., 2013, 2015; Lei et al., 2009; Long et al., 2009; Moustafa et al., 2017; Xu & Chen, 2009). Des hybridations utilisant cette méthode se retrouvent également dans la littérature dont quelques-unes avec la modélisation d'un modèle de Markov caché (Ge & Liang, 2005; Rasmussen & Krink, 2003; J. Sun et al., 2014; J. Sun et al., 2012) et une autre avec la programmation dynamique (Juang & Su, 2008).

Le MSA est un problème complexe à résoudre. Il nécessite l'utilisation de techniques de résolution élaborées avec des stratégies de conception réfléchies pour obtenir des solutions viables. Les métaheuristiques représentent une technique de résolution efficace pour la résolution de ce type de problème (Issa & Hassanien, 2020). La section suivante est dédiée à améliorer la qualité des solutions pour la résolution de ce problème, par la proposition d'un modèle de résolution utilisant le PSO comme stratégie évolutionnaire.

6.2.5. PARAMÈTRES DU MSA POUR LES EXPÉRIMENTATIONS

Les paramètres utilisés pour la résolution du MSA dans ce chapitre correspondent à ceux suggérés par les outils utilisés pour la comparaison des résultats à la Section 6.4.3, c'est-à-dire, ceux pour MAFFT et CLUSTALW. La matrice de substitution sélectionnée est BLOSUM62 et les pénalités d'ouverture et d'extension d'un espacement sont de $\varepsilon = 0{,}5$ et de $\delta = 0{,}1$ respectivement.

La sélection d'une telle matrice et de ces pénalités aide la fonction objectif à fortement pénaliser les ouvertures et à faiblement pénaliser les extensions. Ceci encourage les

espaces singuliers à se regrouper et à créer peu de grands espacements plutôt que plusieurs petits espacements. Une autre raison de l'utilisation de ces paramètres est qu'il s'agit de la plus grande valeur de pénalité d'ouverture ε et de la plus petite valeur de pénalité d'extension δ disponibles et suggérés pour la résolution avec les outils MAFFT. Les paramètres attribués à CLUSTALW utilisent les mêmes que ceux pour MAFFT. Utiliser ainsi ces valeurs permet de comparer la résolution du MSA avec les propositions de ce chapitre en utilisant une base commune.

6.3. PROPOSITION D'UNE NOUVELLE DISCRÉTISATION : CODIFICATION AVEC ANNEAU

6.3.1. REPRÉSENTATION D'UNE SOLUTION

La modélisation de la solution en vue de l'évaluation par la fonction objectif est la même que celle utilisée par Lalwani et al. (2015). Chaque variable de décision correspond à un espace de grandeur 1, représenté dans le vecteur solution avec l'index où est localisé cet espace dans la séquence. Une solution est donc représentée par l'ensemble des index des espaces dans une structure comportant un vecteur de vecteurs. Le premier élément du vecteur contient les espaces de la première séquence de l'alignement, le deuxième élément du vecteur contient les espaces de la deuxième séquence, et ainsi de suite. Plusieurs auteurs utilisent cette structure afin de représenter les solutions à évaluer en utilisant les limites de l'espace de recherche correspondant à 0 et à la longueur maximale autorisée de l'alignement (Mohamed & Aboul Ella, 2017; Moustafa et al., 2017). Un exemple de la structure d'un vecteur solution q comportant quatre séquences est illustré à la Figure 6-6.

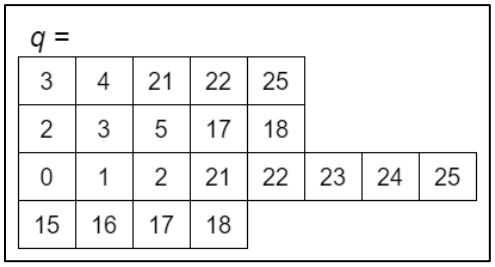

Figure 6-6 : Exemple de codification d'une solution q pour la résolution du MSA.
© Hugo Deschênes

À la figure précédente, le nombre d'espaces à utiliser est 5 pour la première et la deuxième séquence, 8 pour la troisième séquence et 4 pour la quatrième séquence. Avec ces quantités, l'ensemble des séquences possède une longueur L identique correspondant à la longueur de la plus longue séquence, additionnée de 20%. Dans notre exemple, cette valeur L correspond à 26. Les index inscrits à l'intérieur des vecteurs représentent l'emplacement exact de l'espace, où le plus petit index est 0 et le plus grand est $L - 1$. Une telle solution permet, par exemple, de générer un alignement multiple correspondant à la Figure 6-7. Chaque tiret représente un espace ajouté à l'alignement multiple tel que proposé par la solution q de la Figure 6-6.

```
index    0     5     10    15    20    25

séq. 1 : GKG-- DPKKP RGKMS SYAFF V--QR -
séq. 2 : MQ--D -RVKR PMNAF IV--W SRDQR R
séq. 3 : ---MK KLKKH PDFPK KPLTP Y---- -
séq. 4 : MHIKK PLNAF MLYMK ----E MRANV V
```

Figure 6-7 : Exemple d'alignement multiple de séquences avec la solution provenant de la Figure 6-6, avec une longueur maximale $L = 26$.
© Hugo Deschênes

6.3.2. ADAPTATIONS DES PROCÉDÉS DE DISCRÉTISATION POUR LA RÉSOLUTION DU MSA

L'adaptation du CoLPSO pour la résolution du MSA est similaire au fonctionnement du CoLPSO de base de la Section 3.2.3.3. L'algorithme se transpose de la manière suivante.

Chaque sous-essaim (dimension d) représente un espace e à positionner. Chacune des particules x de chaque sous-essaim cherche à positionner adéquatement l'espace au sein du vecteur contextuel cv, lequel représente une solution au problème (l'alignement multiple). Lorsqu'une particule est mise à jour, sa nouvelle position est transmise à cv pour évaluer cet espace avec la fonction objectif. Si cet espace est un duplicata, et donc qu'il est déjà inclus dans cv, cette particule n'est pas évaluée et elle est ignorée pour la génération actuelle. Autrement, elle remplace le dernier espace proposé par le sous-essaim dont elle fait partie. Le vecteur contextuel cv conserve cet espace s'il permet d'améliorer le résultat par la fonction objectif et il rejette cet espace dans le cas inverse.

Il est important de valider la nature idéale des valeurs constituant les composantes d'une particule du PSO. Bien qu'un vecteur solution doit être codifié en valeurs entières, les procédés de discrétisation permettent aux particules d'avoir une codification différente pour la composition de la vélocité et de la position. Il faut alors valider si les performances démontrent un avantage pour l'utilisation de variables discrètes ou plutôt par l'utilisation de variables continues pour ces deux composantes.

6.3.2.1. ADAPTATION DU PROCÉDÉ PAR LA LISTE DE TRANSLATIONS

Les deux procédés de discrétisation proposés dans ce chapitre sont conçus selon le procédé de discrétisation ayant démontré les meilleures performances pour la résolution du SOP et du RCPSP au Chapitre 5, soit la liste de translations (Anghinolfi & Paolucci, 2009). Le première procédé utilise des variables de décision entières pour la composition de la position et de la vélocité d'une particule alors que le deuxième procédé utilise plutôt des variables de décision réelles pour ces mêmes composantes.

Le procédé de discrétisation par liste de translations est défini selon l'utilisation du PSO sous sa version classique. Il utilise l'Équation 5-1 afin de définir la composition de la

vélocité d'une particule. Cette équation représente l'ensemble de toutes les translations (PI) notées $y = (a, \lambda)$ à appliquer aux dimensions d'une particule. Pour cette équation, a représente la variable de décision provenant de l'ensemble A des variables composant le problème combinatoire et λ représente la valeur de la translation. Dans cette version du procédé par liste de translations, un total de Y PI sont incluses dans la vélocité pour une quantité égale à la dimension du problème. Or, chaque particule du CoLPSO est unidimensionnelle, c'est-à-dire qu'elle comporte une position et une vélocité de grandeur 1. Considérant que chaque particule ne représente qu'un seul espace (la variable de décision a), L'Équation 5-1 se simplifie en devenant l'Équation 6-4, où la vélocité n'est composée que d'une seule information : la valeur λ de la translation.

$$v_{id} = \{(a, \lambda)_y \mid y = 1, \dots, Y; a \in A\} \quad (5\text{-}1)$$

$$v_{id} = \lambda \quad (6\text{-}4)$$

Le principal avantage de l'application du procédé de discrétisation par liste de translations au CoLPSO est que cela simplifie la redéfinition des opérateurs arithmétiques identifiés à la Section 5.3.2. En effet, puisqu'il n'y a plus besoin d'une liste de translations considérant l'utilisation de vecteurs unidimensionnels, les opérateurs arithmétiques de l'Équation 3-1 et de l'Équation 3-2 de la mise à jour d'une particule n'ont pas besoin d'être complètement redéfinis. Ceci implique également que le temps de résolution sera inférieur considérant que moins de traitements et de transformations de l'information sont nécessaires pour déplacer les particules. Voici un rappel des quatre opérateurs arithmétiques :

OP-1 La soustraction $(-)$ entre deux vecteurs position ;

OP-2 La multiplication $(\cdot)$ d'un scalaire par une vélocité ;

OP-3 L'addition $(+)$ entre plusieurs vélocités; et

OP-4 L'addition $(+)$ entre une position et une vélocité.

6.3.2.2. UTILISATION DE VARIABLES DE DÉCISION ENTIÈRES

Pour ce qui est de l'utilisation de variables de décision entières, la valeur de translation attribuée à λ représente la valeur entière du déplacement à appliquer à la l'index d'un espace e. Les opérateurs OP-1, OP-3 et OP-4 demeurent identiques puisqu'un entier additionné à un autre entier produit un entier. L'adaptation du procédé concerne l'opérateur OP-2, où le résultat de cette multiplication est redéfini par une troncature des décimales de la multiplication entre une valeur réelle et une valeur entière. Bien qu'il n'y ait pas de limite à la valeur de la vélocité, l'index représenté par la position d'une particule doit être conscrit à l'intérieur de la longueur maximale L de l'alignement multiple. Les limites de l'espace de recherche correspondent donc à 0 et à $L - 1$. Pour le reste de ce chapitre, cette version du procédé de discrétisation est nommée *Liste de Translations – Index (LT-Index)*.

6.3.2.3. UTILISATION DE VARIABLES DE DÉCISION RÉELLES

À propos de l'utilisation de variables de décision réelles, la composition de la position d'une particule représente la proportion de la longueur de la séquence où est localisé un espace. Aucun opérateur arithmétique n'a besoin d'être redéfini. En effet, les données utilisées dans les équations de mise à jour d'une particule sont toutes de nature continue. La valeur de la translation attribuée à λ représente la valeur réelle du déplacement d'un espace e selon le changement de la proportion. En changeant cette proportion, l'espace est déplacé à un autre emplacement (une autre proportion) au sein de l'alignement. Par exemple, pour un alignement comportant une longueur $L = 26$ et une position correspondant à la proportion 0,81, l'espace correspond à l'index 21 ($0{,}81\ x\ L = 21{,}06 = 21$). La position d'une particule a donc besoin de subir une transformation de sa valeur afin de produire une solution selon la représentation décrite à la Section 6.3.1. La Figure 6-8 illustre un exemple de la transformation d'une position x_i codifiée en valeurs réelles en une solution q codifiée en

valeurs entières. Ces deux codifications permettent d'obtenir la même solution présentée à la Figure 6-7, et reproduite à la Figure 6-8.

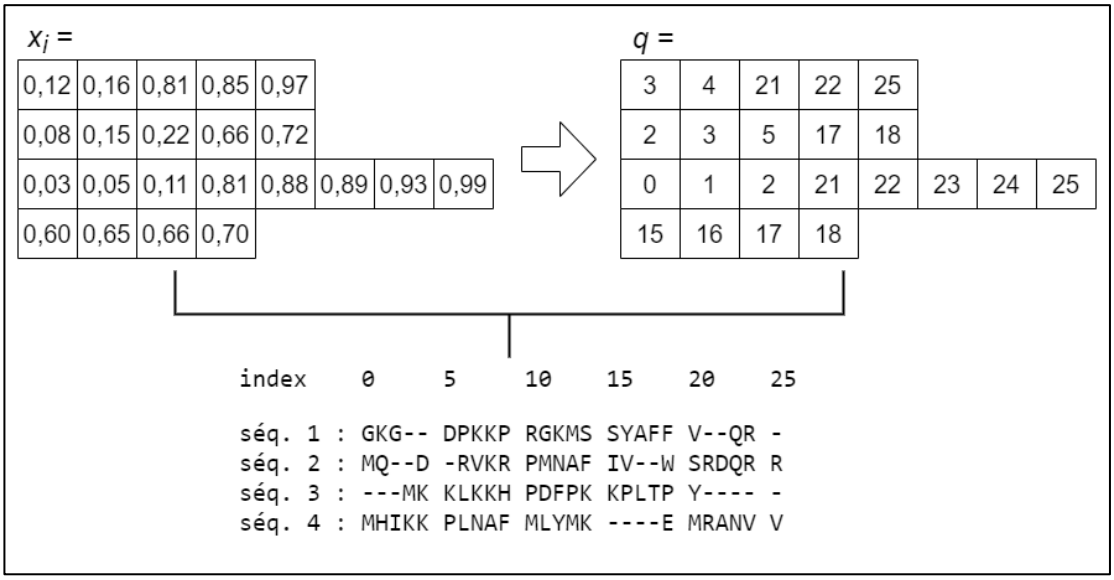

Figure 6-8 : Illustration de la transformation d'une position x_i codifiée en variables réelles pour une solution q codifiée en variables entières.
© Hugo Deschênes

Pour ce procédé de discrétisation, les valeurs limites de l'espace de recherche sont modifiées. Au lieu d'utiliser des valeurs variant de 0 à $L-1$ pour un alignement multiple de grandeur L, celles-ci se limitent plutôt à l'intervalle entre 0,0 et 1.0. La valeur entière obtenue pour construire le vecteur contextuel cv est calculée avec la longueur L, en multipliant la proportion réelle par cette longueur. Une troncature des décimales a lieu sur ce résultat afin d'obtenir un index entier. Pour le reste de ce chapitre, cette discrétisation est nommée *Liste de Translations – Proportion (LT-Proportion)*.

6.3.3. MODÉLISATION DE SOLUTION EN ANNEAU

L'observation des solutions typiques provenant de la résolution d'un alignement multiple de séquences permet de tirer certaines conclusions. Chacune des séquences utilisées dans une instance de problème comporte une longueur différente. Alors qu'une

séquence peut être composée d'une centaine de caractères, une autre peut en contenir une cinquantaine ou moins encore. Ceci crée naturellement une ou plusieurs agglomérations d'espaces (nommés espacements) en début, au milieu et/ou en fin de séquence. Il est en effet plus naturel d'observer une séquence comportant des acides aminés contigus qu'une séquence comportant des acides aminés coupés à maintes reprises par plusieurs espaces singuliers (Pevsner, 2015).

La Figure 6-9 illustre un exemple typique de solution pour l'instance BB11001 fournie avec la banque d'instances BAliBASE 3.0 (Bahr et al., 2001; Thompson et al., 2005). Cette instance comporte quatre séquences (*1aab_*, *1j46_A*, *1k99_A* et *2lef_A*). Dans cette figure, le nom de la séquence de base est défini à gauche et un tiret (-) représente un espace. Il est possible d'observer qu'il y a une agglomération d'espaces en début et en fin de séquence pour plusieurs des séquences de l'instance.

```
1aab_  : ---GKGDPKKPRGKMSSYAFFVQTSREEHKKKHPDASVNFSEFSKKCSERWKTMSAKEKGKFEDMAKADKARYEREMKTYIPPKGE----------
1j46_A : ------MQDRVKRPMNAFIVWSRDQRRKMALENP--RMRNSEISKQLGYQWKMLTEAEKWPFFQEAQKLQAMHREKYPNYKYRPRRKAKMLPK---
1k99_A : MKKLKKHPDFPKKPLTPYFRFFMEKRAKYAKLHP--EMSNLDLTKILSKKYKELPEKKKMKYIQDFQREKQEFERNLARFREDHPDLIQNAKK---
2lef_A : --------MHIKKPLNAFMLYMKEMRANVVAEST--LKESAAINQILGRRWHALSREEQAKYYELARKERQLHMQLYPGWSARDNYGKKKKRKREK
```

Figure 6-9 : Représentation visuelle de la solution proposée pour l'instance BB11001 provenant de *BaliBase3.0* (Bahr et al., 2001; Thompson et al., 2005).

La principale difficulté lors de la résolution du MSA provient de la capacité qu'a l'algorithme à regrouper des espaces entre eux afin de créer des espacements. Ceci permet de diminuer la pénalité totale accordée à l'ouverture des espacements dans une séquence, pour maximiser le score total de l'alignement selon l'évaluation par la fonction objectif (l'Équation 6-3). Il est donc pertinent de concevoir une stratégie permettant d'aider l'algorithme à regrouper les espaces ensemble, et diminuer ainsi les pénalités occasionnées par l'ouverture de plusieurs espacements.

Considérant que le début et la fin de plusieurs séquences dans un alignement multiple sont souvent composés d'espaces, un nouveau comportement est proposé dans ce chapitre pour contribuer à alléger la difficulté à créer, avec une métaheuristique, un regroupement d'espaces en début et en fin de séquence. Il s'agit d'une modélisation de solutions représentée sous la forme d'un anneau. La Figure 6-10 illustre ce concept en utilisant un exemple comportant quatre séquences avec une longueur totale L pour l'ensemble de l'alignement. Les identifiants des colonnes (*Id. Colonne*) sont représentés pour situer le début et la fin de chacune des séquences.

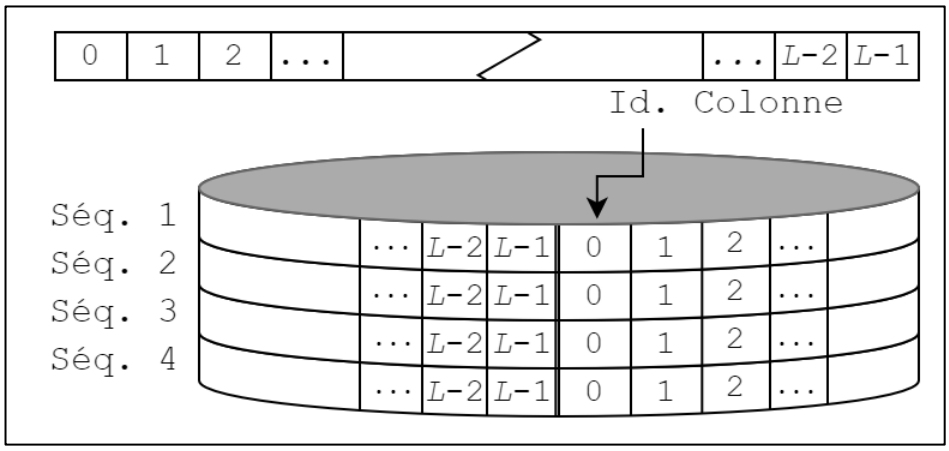

Figure 6-10 : Représentation visuelle d'une solution avec la modélisation en anneau pour la résolution du MSA.
© Hugo Deschênes

La modélisation en anneau représente un avantage pour la constriction des variables de décision à l'intérieur de l'espace de recherche. En effet, beaucoup de problèmes d'optimisation de la littérature comportent des limites de valeur minimale et maximale causées par la nature du problème et des instances utilisées. Ces variables doivent donc être conscrites pour éviter de créer des solutions invalides avec des valeurs extérieures aux limites permises. Avec une modélisation en anneau, cette constriction n'existe plus. Si une donnée excède la valeur maximale d'une unité, elle obtient la valeur minimale de l'espace de recherche, et vice-versa. Ce même principe s'applique autant pour la codification réelle que pour la codification entière. Pour la codification réelle dans un espace de recherche

comportant des valeurs limitées par 0.0 et 1.0, la valeur suivant 1.0 est 0.0 et la valeur précédant 0.0 est 1.0. Pour la codification entière, la valeur suivant $L-1$ est 0 et la valeur précédant 0 est $L-1$. La constriction des variables de décision à l'intérieur de l'espace de recherche et la création potentielle de solutions invalides sont de ce fait allégées. Il n'est donc plus nécessaire de s'en soucier lors de l'adaptation de la métaheuristique au problème.

Dans le cas de l'utilisation du PSO classique, la mise à jour d'une particule implique que celle-ci peut se retrouver au-delà des limites de l'espace de recherche. Une particule qui dépasse ainsi une limite se voit contrainte d'arrêter son mouvement et sa vélocité devient nulle. Avec l'utilisation de l'anneau, la particule n'est pas contrainte dans son mouvement. Elle poursuit son déplacement sans impacter sa vélocité. Cette vélocité est donc conservée et ne vient pas créer un arrêt abrupt d'une particule aux limites de l'espace de recherche. Cet espace devient donc continu. Lors du calcul de la vélocité entre deux particules (et donc du différentiel entre deux positions), la vélocité est calculée en validant le différentiel (la distance) entre les deux positions par la gauche et par la droite. La plus petite distance entre ces deux valeurs est celle sélectionnée pour effectuer la différence. Prenons pour exemple la position d'une dimension pour deux particules $x_{1d}=2$ et $x_{2d}=19$. Considérons également un alignement de longueur $L=20$ avec les valeurs d'index possibles entre 0 et 19. Deux vecteurs différentiels peuvent être calculés : par la droite avec $x_{2d}-x_{1d}=17$, et par la gauche avec $x_{1d}-x_{2d}=-17$. La valeur -17 excédant ainsi l'espace de recherche, elle est ajustée afin de calculer le différentiel en anneau avec le calcul $L+(-17)=20-17=3$. La différence entre x_{2d} et x_{1d} est donc de 3, et cette différence est conservée étant donné que $x_{2d}-x_{1d}>x_{1d}-x_{2d}$, et donc que $17>3$.

Dans le cadre de ce chapitre, la modélisation en anneau est utilisée pour représenter l'espace de recherche et l'évolution des solutions. Les deux procédés de discrétisation (LT-Index et LT-Proportion) sont expérimentés et analysés avec cette modélisation.

6.3.4. BASE COMMUNE DES EXPÉRIMENTATIONS

Suivant les conclusions provenant du Chapitre 4, la variante de PSO permettant d'obtenir les performances les plus intéressantes est le CoLPSO (van den Bergh & Engelbrecht, 2004). En effet, la séparation des dimensions d'une particule en plusieurs sous-essaims aide à obtenir de bonnes performances pour la résolution de problèmes de grande taille en un temps raisonnable. Considérant cet avantage, et par l'utilisation d'une nouvelle modélisation en anneau, des tests sont effectués afin de valider la discrétisation du PSO la plus performante pour résoudre le MSA.

Pour chacune des expérimentations de ce chapitre, le CoLPSO est défini avec les paramètres qui suivent. Après avoir effectué des tests empiriques, une population de 20 particules par sous-essaim a été déterminée, avec un nombre d'évaluations défini à 1 000 fois le nombre correspondant à la dimension de l'instance utilisée. Les paramètres de confiance en soi c_1 et de confiance en la collectivité c_2 sont définis à 1.49445, lequel est un nombre démontré efficace pour le PSO (R. C. Eberhart & Shi, 2000). Le paramètre d'inertie w varie selon la diminution dynamique proposée par Ratnaweera et al. (2004), laquelle correspond à l'utilisation de l'Équation 3-5. Le paramètre w_0 est donc défini à 0,9, et le paramètre w_1 à 0,4.

Tous les tests exécutés pour ce chapitre utilisent un ordinateur comportant le système d'exploitation Windows 10 Professionnel, avec un processeur Intel® Core™ i5-2400 incluant un CPU à 3.10GHz et 8,00 Go de mémoire vive. Chacune des prochaines sections présentant une expérimentation comporte 50 exécutions pour chaque méthode testée pour chacune des instances comportant une dimension inférieure à 1 500. Les résultats sont comparés de manière identique, c'est-à-dire en fonction des moyennes, des médianes, de la meilleure et de la moins bonne valeur selon l'évaluation par la fonction objectif, et ce, pour l'ensemble des exécutions. Les temps d'exécution, la rapidité de convergence et le

pourcentage de particules stagnantes sont trois autres facteurs qui servent à la comparaison selon les sections.

Ces paramètres de comparaison permettent de statuer au cours de ce chapitre sur les différents sélections, paramètres et stratégies démontrant une performance et une amélioration pour l'utilisation du PSO lors de la résolution du problème de support sélectionné, le MSA.

6.3.5. EXPÉRIMENTATIONS SUR LES PROCÉDÉS DE DISCRÉTISATION

Des tests sont réalisés afin de comparer la performance du CoLPSO selon les deux procédés de discrétisation décrits à la Section 6.3.2, soit LT-Index et LT-Proportion. L'utilisation de la modélisation de solutions en anneau est utilisée pour ces deux procédés.

Le Tableau 6-2 présente les résultats provenant des tests exécutés sur l'ensemble des 27 instances sélectionnées et décrites à la Section 6.2.3. L'accent est mis sur la différence entre les performances de LT-Index et LT-Proportion, sans chercher, pour le moment, à résoudre le problème de manière compétitive avec les outils du marché. Les instances sont également présentées en ordre croissant de dimension. La Figure 6-11 présente les mêmes résultats qu'au Tableau 6-2, illustrant plutôt les résultats empiriques de la résolution des instances selon la dimension.

Tableau 6-2 : Résultats empiriques pour la résolution du MSA avec les deux procédés de discrétisation LT-Index et LT-Proportion

Instance	BB11001		BB11025		BB11021		BB11013		BB11029		BB11017		BB11015		BB11035		BB11010	
Dimension	91		161		195		247		252		253		334		338		394	
Proc. de discr.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	Lt-Index	LT-Prop.
Min	***40,0***	-56,6	***-126,1***	-307,7	***-73,6***	-341,2	***-1,3***	-360,7	***-61,4***	-344,6	***-159,1***	-431,8	***2,0***	-471,7	***94,6***	-370,0	***-596,6***	-1010,7
Max	***296,2***	122,8	***78,9***	-109,7	***233,5***	-50,2	***183,8***	-133,4	***159,7***	-145,2	***329,8***	162,9	***707,5***	181,6	***410,7***	-68,2	***42,6***	-391,4
Moyenne	***143,2***	40,9	***-21,3***	-194,9	***96,2***	-173,7	***80,2***	-231,6	***53,7***	-239,5	***95,4***	-182,5	***315,4***	-71,0	***281,7***	-219,1	***-309,8***	-730,4
Médiane	***140,0***	43,2	***-12,9***	-190,6	***92,4***	-171,3	***85,6***	-228,1	***54,1***	-246,4	***86,1***	-194,7	***306,2***	-69,7	***298,8***	-225,4	***-320,8***	-710,5
Tps. Exéc.	***2,1***	2,4	***3,4***	4,7	***6,2***	9,0	***6,6***	11,0	***6,5***	11,4	***21,1***	27,0	***33,5***	44,7	***13,1***	22,8	***64,5***	81,7
Instance	BB11011		BB11012		BB11004		BB11022		BB11020		BB11003		BB11024		BB11009		BB11006	
Dimension	445		448		504		512		583		596		604		752		754	
Proc. de discr.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	Lt-Index	LT-Prop.
Min	***-133,2***	-700,8	***-68,9***	-678,8	***-388,4***	-955,7	***-35,3***	-572,5	***-270,8***	-1171,9	***-299,5***	-1123,3	***-350,9***	-1070,2	***-148,6***	-970,5	***-337,1***	-1591,5
Max	***329,1***	-280,5	***676,2***	-4,1	***72,0***	-560,6	***135,0***	-365,1	***1043,5***	36,2	***157,6***	-581,1	***171,6***	-680,1	***156,9***	-626,9	***661,1***	-829,1
Moyenne	***113,3***	-495,9	***178,6***	-400,5	***-159,3***	-733,7	***54,3***	-459,7	***523,5***	-589,2	***-66,3***	-801,5	***-156,1***	-839,8	***20,2***	-799,3	***126,1***	-1155,2
Médiane	***130,0***	-516,6	***175,3***	-411,1	***-155,3***	-743,1	***53,8***	-465,6	***523,4***	-588,4	***-62,0***	-795,4	***-145,2***	-843,9	***24,6***	-797,1	***117,0***	-1126,5
Tps. Exéc.	***40,1***	60,5	***52,4***	74,1	***68,3***	96,2	***13,2***	38,5	***182,4***	228,9	***86,4***	128,4	***77,3***	117,8	***32,4***	97,4	***198,1***	274,6
Instance	BB11028		BB11002		BB11007		BB11036		BB11023		BB11019		BB11014		BB11008		BB11033	
Dimension	942		1064		1167		1224		1247		1266		1323		1372		1482	
Proc. de discr.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	LT-Index	LT-Prop.	Lt-Index	LT-Prop.
Min	***564,0***	-1258,7	***496,6***	-1295,8	***889,2***	-1514,2	***761,8***	-1598,1	***860,8***	-1288,3	***1413,0***	-1546,9	***325,4***	-1676,8	***-94,8***	-1294,8	***1482,4***	-1429,7
Max	***1383,7***	-473,7	***896,6***	-665,2	***3060,4***	721,3	***1890,9***	-394,4	***2101,6***	-72,1	***3578,0***	1086,6	***2157,4***	-382,0	***155,1***	-1009,5	***2607,6***	-458,8
Moyenne	***948,4***	-932,9	***703,0***	-968,3	***1793,7***	-634,8	***1230,5***	-933,4	***1478,1***	-639,7	***2606,8***	-236,0	***1117,2***	-1093,7	***39,6***	-1140,7	***2077,8***	-949,8
Médiane	***935,7***	-950,7	***692,2***	-971,1	***1720,1***	-629,8	***1227,5***	-912,0	***1496,0***	-650,4	***2633,2***	-241,4	***1073,4***	-1102,9	***42,7***	-1138,7	***2073,2***	-974,9
Tps. Exéc.	***206,2***	332,7	***81,8***	193,3	***668,4***	895,4	***468,8***	691,2	***286,3***	490,7	***709,0***	935,3	***438,0***	637,3	***70,1***	275,5	***324,8***	572,5

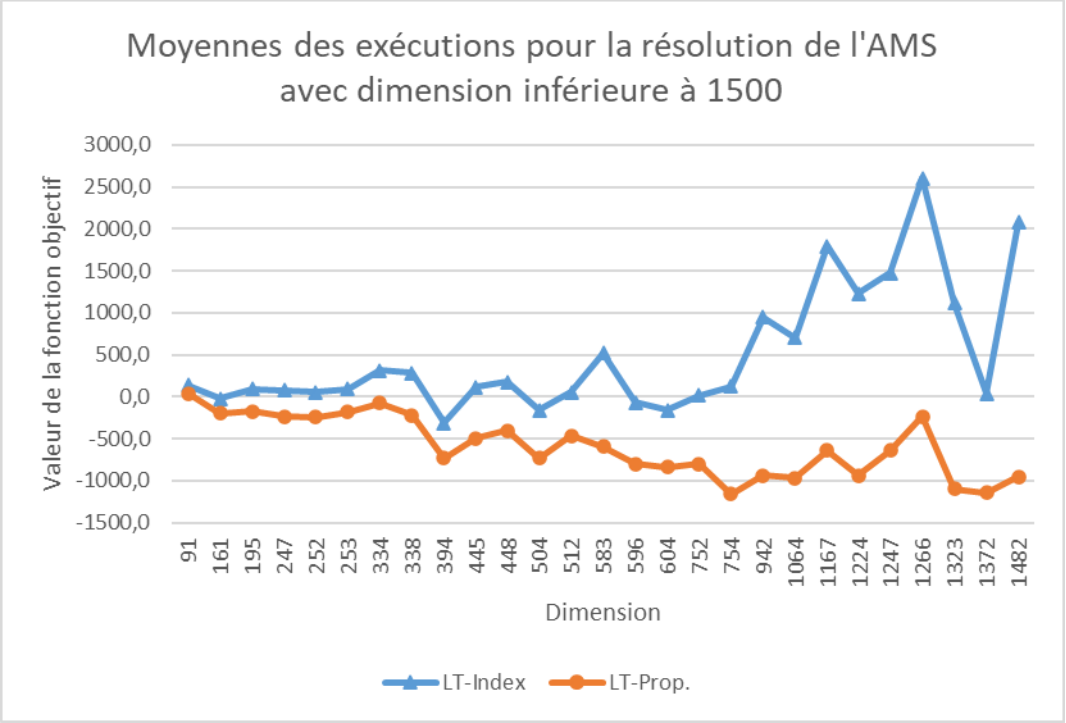

Figure 6-11 : Illustration de la résolution des instances de MSA en comparant l'utilisation des variables entières (LT-Index) et des variables réelles (LT-Proportion)
© Hugo Deschênes

Selon ce tableau et cette figure, les tests permettent d'observer que la résolution par l'utilisation de variables entières (LT-Index) donne des résultats supérieurs à l'utilisation des variables réelles (LT-Proportion). En effet, le procédé de discrétisation LT-Index surpasse le procédé de discrétisation LT-Proportion sur l'ensemble des 27 instances, et ce, autant pour les meilleurs minimums, maximums, moyennes, médianes et temps d'exécution. Par exemple, l'utilisation de variables entières pour la plus petite instance (BB11001) obtient une moyenne et une médiane de 296,2 et de 143,2 respectivement, en comparaison avec l'utilisation de variables réelles qui obtient 40,9 et 43,2 pour les mêmes données statistiques. Pour la plus grande instance (BB11033), l'utilisation des variables entières obtient une moyenne et une médiane de 2077,8 et de 2073,2 respectivement, en comparaison avec l'utilisation de variables réelles qui obtient -949,8 et -974,9 pour les mêmes données statistiques. Il semble ressortir que la discrétisation LT-Index apporte une amélioration de la

qualité des solutions dans la résolution des instances de MSA comportant une dimension en inférieure à 1 500.

Un autre élément de comparaison servant à valider le procédé de discrétisation le plus performant concerne la rapidité de convergence des particules de l'essaim. En effet, la rapidité avec laquelle un algorithme atteint la meilleure solution pour une exécution est un élément permettant de mesurer l'efficacité qu'a un algorithme à résoudre une instance.

La Figure 6-12 illustre la moyenne des solutions obtenues selon le nombre d'évaluations effectuées avec la fonction objectif sur l'ensemble des 50 exécutions. Quatre instances de tailles différentes sont utilisées pour la comparaison, soient BB11001, BB11002, BB11004 et BB11033. Ces instances représentent des exécutions typiques pour les instances de MSA comportant une dimension définie respectivement à 91, 504, 1064 et 1482. Cette figure permet d'observer la rapidité de la convergence de l'ensemble des particules de l'hybride HCoCLPSO lors de la résolution du MSA.

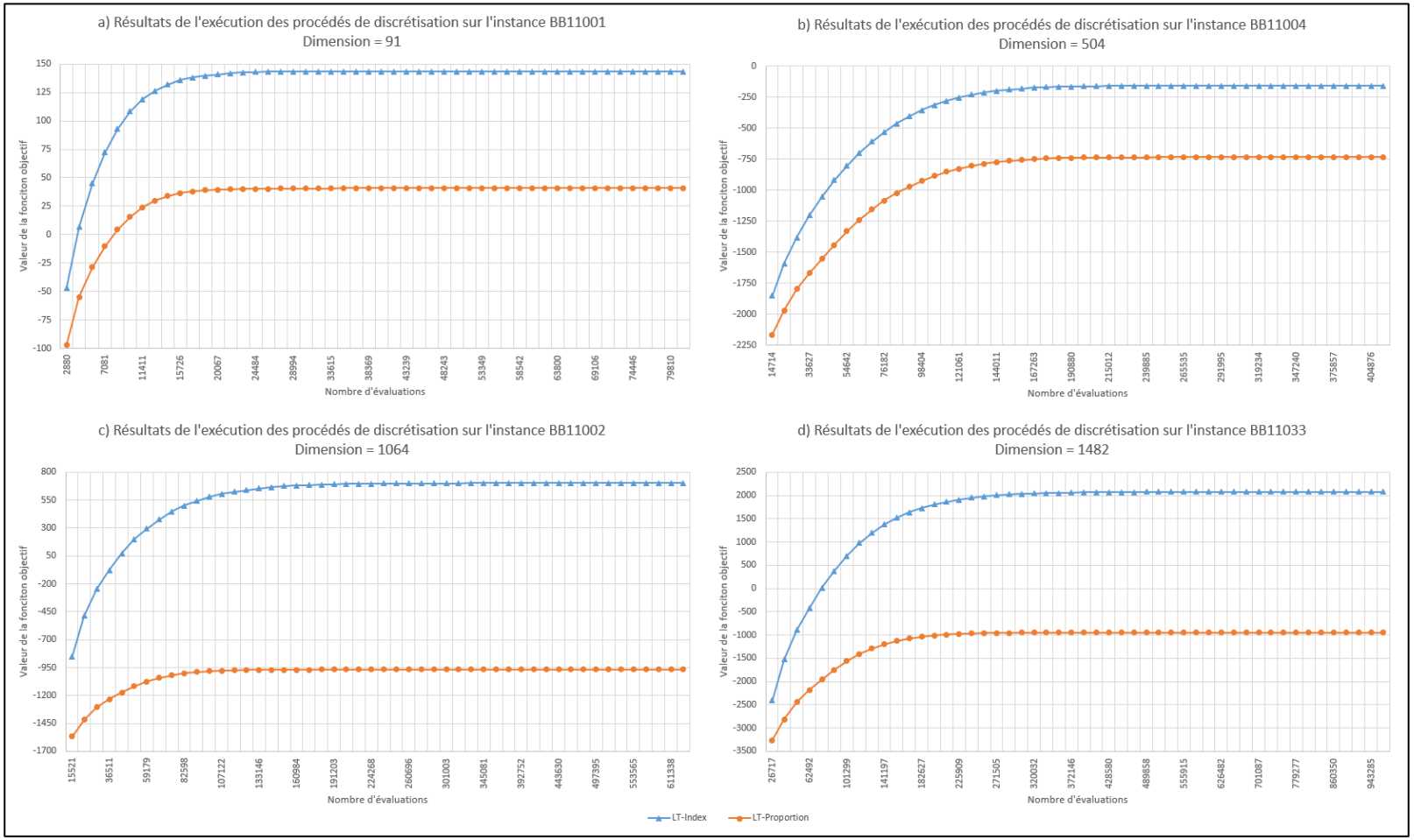

Figure 6-12 : Représentation visuelle de la convergence des particules vers la moyenne des solutions obtenues pour les procédés de discrétisation LT-Index et LT-Proportion, selon la résolution des instances en a) BB11001, b) BB11004, c) BB11002 et d) BB11033.

La Figure 6-12 permet d'observer la rapidité de la convergence des particules du CoLPSO pour le procédé de discrétisation LT-Index (points triangulaires) et LT-Proportion (points circulaires). Plus spécifiquement, elles permettent d'observer que les deux procédés convergent rapidement vers une solution et n'utilisent pas l'entièreté des évaluations permises avec la fonction objectif pour continuer d'améliorer les solutions. En effet, les meilleures solutions obtenues pour les instances typiques ne changent pas ou très peu après avoir utilisé près de la moitié des évaluations de solutions octroyées, et ce, pour les deux procédés de discrétisation. La convergence est alors très rapide. En contrepartie, ceci signifie également qu'il y a encore beaucoup d'effort de calculs qu'il est possible d'utiliser pour l'ajout de processus complémentaires permettant de poursuivre l'amélioration des solutions.

Ces tests permettent de statuer que la codification la plus performante pour les valeurs composant la position et la vélocité d'une particule est de type entier avec une codification des solutions en anneau. Les performances de l'anneau dépendent de la manière dont il est utilisé. En effet, l'anneau améliore la malléabilité du PSO et augmente les possibilités d'adaptation de cette métaheuristique pour la conception d'un algorithme hybride. Il est possible d'utiliser cette liaison entre le début et la fin d'une solution du MSA pour poursuivre l'amélioration des solutions et concevoir de nouvelles stratégies de résolution.

6.4. PROPOSITION D'UN NOUVEAU PROCESSUS D'APPRENTISSAGE COMPRÉHENSIF : LE MAGNÉTISME

Avec les conclusions de la section précédente, il est déterminé que la codification la plus performante des particules consiste à utiliser les variables de décision entières (LT-Index) avec la modélisation de solutions en anneau. Suivant cette conclusion, la suite des expérimentations implique l'hybridation. Tel que spécifié à la Section 2.2.3, la performance d'une métaheuristique est fortement tributaire de son adaptation pour résoudre un problème.

Pour ce faire, la méthode hybride HCoCLPSO est utilisée considérant qu'il s'agit de l'hybridation ayant obtenu les meilleures performances au Chapitre 4 pour la résolution du SOP et du RCPSP. Une nouvelle conception de cet hybride est proposée dans cette section.

La particularité de l'hybride HCoCLPSO est l'ajout du processus d'apprentissage compréhensif provenant du CLPSO au début de la mise à jour des sous-essaims du CoLPSO. Ce processus d'apprentissage compréhensif cherche à améliorer l'exploration de l'espace de recherche lorsque les solutions stagnent pendant un trop grand nombre de générations. Tel qu'il a été démontré à la Figure 6-12, les particules stagnent rapidement alors que moins de la moitié des évaluations sont utilisées pendant la résolution des instances de problème. Le processus d'apprentissage compréhensif est le médium ayant permis d'améliorer les solutions au Chapitre 4 lors de la résolution du SOP et du RCPSP. Il s'agit également du médium adapté et redéfini permettant de concevoir un hybride performant pour la résolution du MSA. La modélisation d'une solution en anneau vient aussi supporter cette nouvelle proposition.

6.4.1. PRÉSENTATION DU MAGNÉTISME

Un nouveau processus d'apprentissage, nommé *Magnet*, est présenté dans cette section. Ce nouveau processus vise à aider le HCoCLPSO à résoudre les instances de MSA de grande taille. Il est noté à la Section 6.2 qu'un comportement régulier observable dans les solutions du MSA consiste à obtenir des espaces regroupés, minimisant ainsi le nombre d'ouvertures d'espacements. En effet, un petit nombre d'ouvertures contribue à maximiser la valeur de la fonction objectif en diminuant le total des pénalités occasionnées avec l'Équation 6-2. Selon les tendances provenant de la littérature, la valeur affectée à la pénalité d'ouverture ϵ est plus grande que la valeur affectée à la pénalité d'extension δ. Le nouveau processus Magnet vise à optimiser le MSA en contribuant à regrouper les espacements et

ainsi diminuer le total des pénalités. Le nouvel hybride créé par cette adaptation se nomme HCoCLPSO-Magnet.

Le nouveau processus d'apprentissage Magnet fonctionne de manière à reproduire le comportement du magnétisme entre un aimant et un matériau ferromagnétique. Au début de chaque itération de l'algorithme, la solution contenue à l'intérieur du vecteur contextuel cv subit le processus d'apprentissage. Celui-ci applique à cv une série d'améliorations cherchant à regrouper les espacements ensemble. Pour ce faire, un espacement aléatoire est sélectionné dans chacune des séquences de l'alignement, puis concaténé aléatoirement entre le prochain espacement situé à droite et le prochain espacement situé à gauche au sein de la même séquence. Avec la modélisation en anneau, ceci implique que l'espacement à la droite du dernier espacement dans une séquence est le premier, et vice-versa. Ceci permet d'aider ces espacements à se regrouper et à créer un grand espacement en début/fin de séquence. Si le déplacement effectué contribue à améliorer le score de l'alignement des acides aminés dans les colonnes impliquées par ce changement, la modification est conservée. Le processus Magnet effectue ensuite un appel récursif en recommençant ce processus une nouvelle fois avec l'utilisation de la solution améliorée. Autrement, si le déplacement effectué diminue le score de l'alignement, le magnétisme est rejeté et un autre espacement aléatoire est sélectionné pour effectuer une autre magnétisation.

La Figure 6-13 illustre un exemple d'itération de magnétisme appliquée sur un espacement d'une séquence provenant d'un alignement multiple. Dans cet exemple, le magnétisme permet de déplacer un espacement de longueur 3 afin qu'il soit magnétisé à l'espace le plus près localisé à gauche en A, ou bien à l'espace le plus près localisé à droite en B. En A, le magnétisme et la modélisation en anneau permettent d'obtenir un espacement de longueur 17 au lieu de 14, alors qu'en B, la longueur de l'espacement est augmentée à 5 au lieu de 2. Dans les deux cas, le nombre total d'ouverture d'espacements pour la séquence diminue de 1.

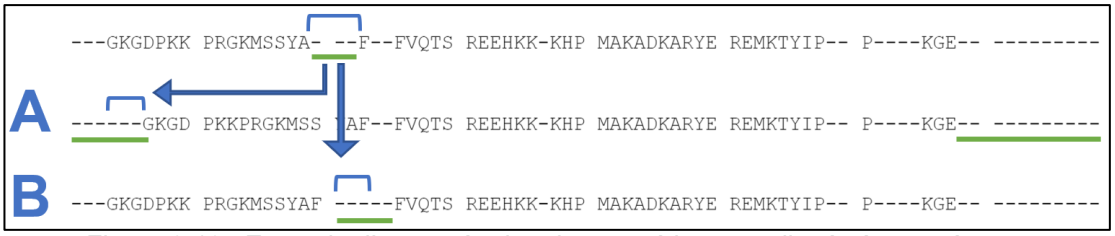

Figure 6-13 : Exemple d'une opération de magnétisme appliquée à une séquence d'un alignement.
© Hugo Deschênes

La Figure 6-14 illustre l'organigramme du HCoCLPSO-Magnet. Le nouveau processus d'apprentissage Magnet est enclenché en ③ avant la mise à jour des D dimensions de l'instance. La mise à jour de ces dimensions est identique à celle provenant du CoLPSO avec l'encadré 3-7D provenant de la Figure 3-7.

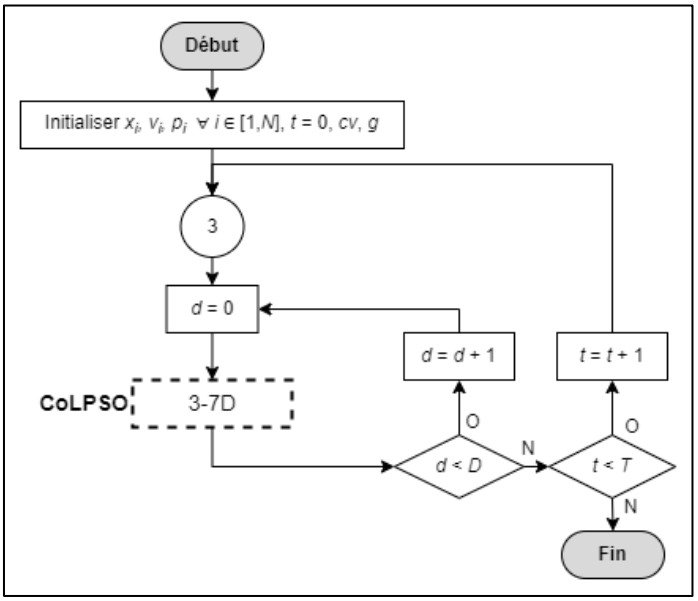

Figure 6-14 : Organigramme du HCoCLPSO-Magnet.
© Hugo Deschênes

Avant d'amorcer les explications du nouveau processus Magnet en ③, définissons les variables permettant son exécution. Il est important de spécifier que ce processus utilise une méthode récursive afin d'améliorer les solutions. La variable $gén$ sert à valider la profondeur de la récursivité en évitant, pour le processus entier, d'aller plus profond que la 20e génération. Cette variable est incrémentée de 1 à l'entrée de la méthode récursive et elle est

diminuée de 1 avant sa sortie. Ceci implique que si le processus utilise les appels récursifs pour un total de 20 fois, il cessera la recherche en profondeur. Cette limite de 20 a été déterminée à l'aide de tests empiriques. Également, la variable cpt est utilisée afin de valider le nombre de fois où l'opération de magnétisme est effectuée sur une même génération $gén$. Une variable s sert quant à elle à connaître le numéro de la séquence en traitement parmi les S séquences de l'alignement. Ensuite, la variable $nbAlea$ contient le résultat du tirage d'un nombre aléatoire. Finalement, la variable γ sert à compter le nombre de fois où l'opération de magnétisme entre deux espacements est effectuée pendant l'entièreté du processus. Pour limiter le nombre de modifications de magnétisme, un nouveau paramètre Γ est défini. Celui-ci correspond au nombre maximal d'appels au magnétisme. La valeur idéale pour Γ constitue le sujet de la Section 6.4.2.

Quatre fonctions sont utilisées. La première, nommée $f(q)$, effectue l'évaluation d'une solution q par la fonction objectif $f()$ définie à la Section 6.2.2.3. La deuxième, nommée $eval(q)$, évalue l'impact du processus Magnet sur la solution q en évaluant seulement les colonnes modifiées avec le processus. La solution n'est donc pas évaluée dans son intégralité. La troisième, nommée $nbEsp(q)$, compte le nombre total d'espacements dans la solution q pour l'ensemble de l'alignement. La quatrième, nommée $alea(nb1, nb2)$, détermine un nombre aléatoire entre $nb1$ et $nb2$ inclusivement, selon la nature entière ou réelle des paramètres.

Différentes structures de données sont également définies, lesquelles représentent pour la plupart une solution potentielle au problème. Parmi celles-ci, $initiale$, $meilleure$, $début$ et $base$ sont initialisées avec une copie du vecteur contextuel cv. La solution $initiale$ représente la solution au commencement du processus Magnet et la solution $meilleure$ représente la meilleure solution obtenue pendant l'application du processus. La solution $début$ représente la solution à l'entrée de la méthode récursive de magnétisme à l'encadré 6-16-MR (Figure 6-16) et la solution $base$ représente la solution au début des changements

apportés avec les opérations de magnétisme à l'encadré 6-16-M (Figure 6-16). Une autre structure de données, nommée ve, est un vecteur contenant l'ensemble des espacements e appartenant à une séquence s du vecteur contextuel cv. Le nombre total d'espacements pour une séquence correspond à E.

L'exécution du nouveau processus Magnet enclenché en ③ à la Figure 6-14 est détaillé à la Figure 6-15 et à la Figure 6-16. Il débute avec l'initialisation des variables γ et $gén$, de même qu'avec l'initialisation des solutions $initiale$ et $meilleure$. Par la suite, la méthode récursive de magnétisme est exécutée à l'encadré 6-16-MR. Une fois cette méthode complétée, il y a une comparaison entre la solution *initiale* et la solution *meilleure* afin de valider laquelle permet d'obtenir la meilleure valeur selon l'évaluation par la fonction objectif $f(\,)$.

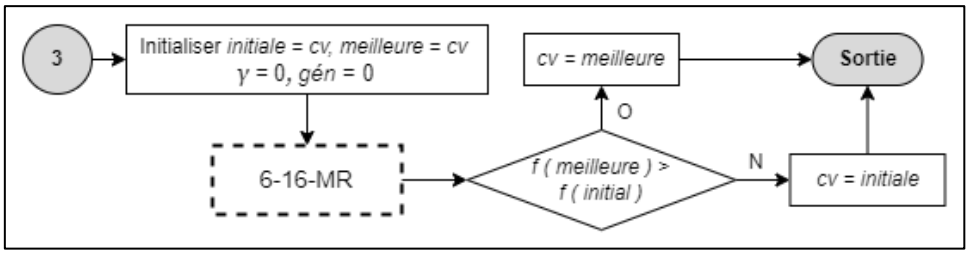

Figure 6-15 : Organigramme du nouveau processus d'apprentissage compréhensif Magnet.
© Hugo Deschênes

L'encadré 6-16-MR (6-16-*Magnétisme Récursif*) est illustré à la Figure 6-16. Il englobe la méthode récursive de magnétisme. Il débute avec l'initialisation du compteur cpt, l'incrémentation du nombre de générations $gén$ et la copie du vecteur contextuel cv dans la solution $début$. S'en suit une boucle de type *TANT QUE* permettant de valider si les trois conditions sont remplies pour sortir de la génération actuelle de la méthode récursive :

1- Si cpt atteint la valeur 10 000 ;

2- Si γ atteint la limite de magnétisme Γ ; ou

3- Si $gén$ atteint la valeur 20.

La satisfaction de ces trois conditions permet de débuter les opérations de magnétisme avec l'encadré 6-15-M. Autrement, $gén$ est diminué de 1 et l'exécution quitte la méthode récursive actuelle afin de remonter et poursuivre l'exécution à la génération précédente.

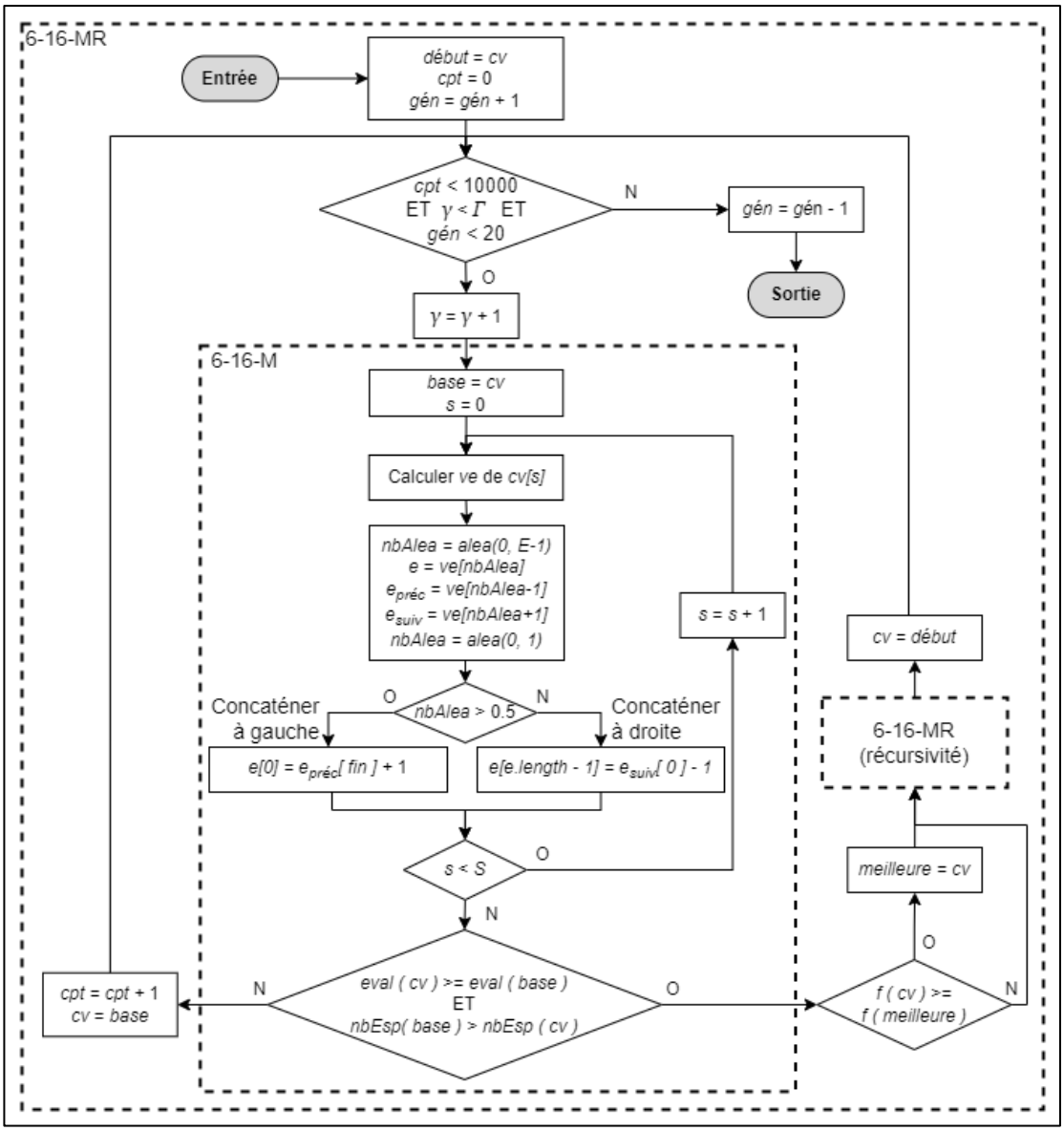

Figure 6-16 : Organigramme du fonctionnement de l'encadré 6-16-MR illustré à la Figure 6-15.
© Hugo Deschênes

L'encadré 6-16-M (6-16-*Magnétisme*) englobe les opérations de magnétisme à effectuer afin de concaténer un espacement avec son voisin de gauche ou de droite. Pour ce faire, l'encadré débute avec l'initialisation de la solution $base$. Le vecteur d'espacements ve est ensuite construit selon les espacements présents dans la séquence s, en débutant avec la première séquence ($s = 0$). Un espacement aléatoire e est sélectionné par la suite et les deux espacements voisins de celui-ci sont identifiés. Selon le tirage d'un nombre aléatoire $nbAlea$ entre 0 et 1, l'espacement e est concaténé à son voisin de gauche (si $nbAlea$ est plus petit que 0,5) ou à son voisin de droite (si $nbAlea$ est plus grand que 0,5). Un espacement est magnétisé de cette manière pour chacune des séquences s de la solution. L'encadré 6-15-M termine avec l'évaluation des changements apportés par le magnétisme avec les fonctions $eval$ et $nbEsp$, en effectuant une comparaison entre la nouvelle solution produite au sein de cv et la solution $base$. S'il y a une amélioration de la qualité de la solution, la nouvelle solution produite est comparée avec la solution $meilleure$ avant d'entrer dans une génération plus profonde dans la récursivité. Autrement, le compteur cpt est incrémenté de 1 et le vecteur cv est réinitialisé avec une copie du contenu du vecteur $base$.

Le nouveau processus Magnet n'utilise pas la probabilité d'apprentissage Pc_i du CLPSO pour enclencher le processus d'apprentissage. Plusieurs raisons expliquent le choix de son omission. Tout d'abord, des tests empiriques ont permis de déterminer que l'utilisation de ce paramètre produit de moins bons résultats lorsqu'il est utilisé. Considérant que les instances de MSA sont particulièrement grandes et qu'il est difficile de regrouper les espacements, le but visé est de débuter le processus d'apprentissage sur le vecteur contextuel cv le plus tôt possible. Il n'y a donc pas l'attente de stagnation avant d'effectuer les opérations de magnétisme. Également, les probabilités d'apprentissage du CLPSO sont calculées et attribuées aux particules. Or, le nouveau processus Magnet est appliqué au vecteur contextuel cv et non à des particules individuelles. Pour ces raisons, l'utilisation de Pc_i devient incohérente avec la stratégie d'hybridation utilisée.

6.4.2. EXPÉRIMENTATIONS POUR LA CONFIGURATION DU MAGNÉTISME

La paramétrisation du paramètre Γ, lequel correspond au nombre maximal d'appels au magnétisme, constitue le sujet des expérimentations de cette section. Pour établir sa valeur idéale, deux techniques d'attribution servant à définir une valeur à Γ sont testées : l'attribution d'une valeur statique (magnétisme statique) et l'attribution d'une valeur dynamique (magnétisme dynamique). Pour définir le paramètre idéal, les tests de cette section utilisent les mêmes bases expérimentales que celles définies à la Section 6.3.4.

6.4.2.1. MAGNÉTISME STATIQUE

La première partie de ces tests correspond à l'attribution des valeurs statiques suivantes : 1 000, 5 000, 10 000 et 20 000. Ces valeurs sont sélectionnées afin d'avoir une idée de la performance du nouveau processus d'apprentissage Magnet selon l'augmentation du nombre d'opérations de magnétisme.

Les tableaux de résultats de cette section présentent la moyenne des 50 exécutions par instance pour les quatre valeurs de Γ définies ci-haut. Étant donné la quantité de données à présenter, les résultats sont divisés selon deux tableaux. Le Tableau 6-3 présente les résultats de la résolution avec les valeurs statiques de Γ pour les instances comportant une dimension inférieure à 600. Le Tableau 6-4 présente à son tour l'équivalent pour les instances comportant une dimension entre 600 et 1 500. Les données en gras italique correspondent au meilleur résultat obtenu parmi les valeurs de Γ selon l'instance et la statistique de minimum (Min), maximum (Max), moyenne, médiane et temps d'exécution (Tps. Exéc.). Une nouvelle donnée statistique est ajoutée pour la comparaison de cette section : le nombre total moyen d'espacements dans la solution obtenue (Total Esp.).

Tableau 6-3 : Résultats de la résolution du MSA avec la configuration du *Magnet* utilisant les valeurs statiques de Γ sur les instances comportant une dimension inférieure à 600.

Instance	BB11001				BB11025				BB11021				BB11013				BB11029			
Dimension	91				161				195				247				252			
Valeur de Γ	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1000	5000	10000	20000
Min	58,9	77,3	***115,1***	78,1	***-40,1***	-75,6	-46,2	-78,3	41,5	76,9	***77,5***	-3,0	-20,3	13,6	***36,0***	22,2	***64,6***	5,0	30,7	48,3
Max	400,8	***427,3***	426,3	422,3	127,4	130,0	***147,0***	90,3	***308,7***	306,6	306,7	307,6	201,5	***275,4***	265,2	244,4	262,6	259,1	272,5	***334,0***
Moyenne	223,3	270,1	266,3	***290,8***	33,3	34,1	***40,9***	20,5	175,8	***206,2***	199,2	172,4	116,4	***136,1***	131,6	128,4	158,2	150,8	146,0	***159,1***
Médiane	223,0	281,4	268,7	***310,9***	35,1	30,0	***36,4***	25,4	174,9	***208,5***	198,9	166,4	123,3	123,9	***133,6***	128,0	***156,9***	141,7	140,9	151,0
Tps. Exéc.	***4,5***	9,1	15,2	52,3	***5,6***	9,7	15,5	51,1	***9,1***	14,4	22,1	70,5	***9,3***	14,8	22,2	69,1	***8,9***	13,4	20,0	63,9
Total Esp.	39	24	***19***	22	41	46	53	***31***	71	57	67	***54***	80	***54***	55	56	57	56	51	***47***

Instance	BB11017				BB11015				BB11035				BB11010				BB11011			
Dimension	253				334				338				394				445			
Valeur de Γ	1000	5000	10000	20000	1 000	5 000	10 000	20 000	1000	5000	10000	20000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000
Min	18,8	106,1	124,2	***126,8***	173,7	213,4	192,5	***264,0***	176,0	***218,2***	174,3	202,3	-450,3	-363,5	-339,4	***-253,8***	-101,6	***114,7***	-19,6	42,9
Max	568,4	640,7	688,7	***693,3***	958,4	***1145,3***	1073,9	1072,3	478,7	***571,2***	522,0	536,4	67,5	***484,6***	463,3	447,1	302,1	410,7	407,7	***520,2***
Moyenne	305,9	359,6	395,7	***430,7***	496,8	644,7	657,3	***673,6***	335,7	381,4	364,3	***382,5***	-165,8	-45,1	17,7	***39,2***	142,8	246,9	234,6	***265,1***
Médiane	308,6	349,2	387,0	***433,6***	469,8	677,4	655,3	***696,3***	332,9	375,8	374,3	***389,1***	-173,4	-63,0	***33,2***	21,5	134,0	242,4	249,8	***265,8***
Tps. Exéc.	***26,3***	35,2	48,3	146,2	***38,3***	49,9	65,5	187,9	***16,2***	22,9	31,9	94,4	***73,0***	89,3	113,2	303,1	***45,2***	57,4	74,0	195,8
Total Esp.	109	***75***	82	83	155	140	121	***108***	104	***64***	80	87	213	182	167	***143***	180	***152***	157	***152***

Instance	BB11012				BB11004				BB11022				BB11020				BB11003			
Dimension	448				504				512				583				596			
Valeur de Γ	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000
Min	132,3	277,4	316,1	***526,9***	-255,0	-130,2	***-52,8***	-65,7	-57,0	-16,4	***37,2***	-8,8	-197,7	***74,7***	-26,6	-82,3	-164,2	-40,8	-102,1	***46,5***
Max	727,5	1189,8	1229,2	***1547,7***	166,4	281,1	***369,6***	352,4	215,6	221,4	***232,6***	225,1	***1348,7***	1063,1	1133,0	1292,7	225,1	387,6	370,9	***421,8***
Moyenne	466,6	710,6	740,7	***864,5***	-8,7	82,4	133,3	***137,4***	109,7	***114,8***	111,0	107,4	609,0	563,5	587,3	***654,1***	48,7	181,1	176,9	***248,0***
Médiane	462,7	692,6	712,3	***842,3***	-3,2	109,9	134,9	***149,4***	108,3	114,4	***115,8***	114,6	602,0	588,8	630,0	***684,7***	51,2	176,9	190,1	***233,9***
Tps. Exéc.	***58,4***	69,2	87,1	242,7	***74,7***	88,8	109,7	282,3	***14,5***	20,1	28,2	86,3	***202,9***	248,3	306,1	680,6	***92,8***	108,2	130,0	325,7
Total Esp.	188	137	127	***107***	227	193	192	***179***	76	***51***	64	57	371	381	356	***318***	275	230	223	***185***

Tableau 6-4 : Résultats de la résolution du MSA avec la configuration du *Magnet* utilisant les valeurs statiques de Γ sur les instances comportant une dimension entre 600 et 1 500.

Instance	BB11024				BB11009				BB11006				BB11028				BB11002			
Dimension	604				752				754				942				1064			
Valeur de Γ	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000
Min	-233,8	***-51,3***	-57,7	-76,8	-15,3	21,3	14,4	***28,2***	-190,6	-178,9	-286,4	***-141,9***	530,9	458,0	414,1	***598,0***	443,9	545,2	***599,7***	597,9
Max	214,4	293,1	316,6	***430,3***	239,8	212,6	256,7	***265,6***	727,0	759,4	734,0	***857,5***	***1862,5***	1699,5	1697,7	1698,0	982,7	1118,4	1091,6	***1131,6***
Moyenne	10,2	105,7	129,5	***184,1***	99,0	135,6	137,8	***160,6***	184,2	320,0	287,5	***378,6***	1017,7	1070,3	1013,4	***1108,9***	765,7	814,3	800,4	***847,4***
Médiane	10,8	108,1	137,3	***174,3***	99,7	143,1	144,4	***156,2***	162,1	316,0	264,9	***394,5***	1020,3	1024,3	975,6	***1109,2***	765,8	805,2	800,0	***845,2***
Tps. Exéc.	***81,5***	94,8	114,9	286,8	***33,5***	41,6	53,6	152,5	***214,0***	252,8	303,2	650,3	***221,1***	261,1	310,7	648,3	***85,7***	101,6	123,0	273,2
Total Esp.	240	223	194	***161***	122	119	125	***96***	417	372	395	***369***	400	404	386	***365***	276	252	233	***219***
Instance	BB11007				BB11036				BB11023				BB11019				BB11014			
Dimension	1167				1224				1247				1266				1323			
Valeur de Γ	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000
Min	***1068,4***	972,8	652,1	1038,1	816,2	921,1	960,5	***966,1***	1185,4	1200,8	***1319,2***	1201,3	1922,4	2056,5	1991,5	***2296,8***	779,3	737,3	841,0	***1234,6***
Max	2893,5	3177,6	3025,5	***3245,1***	1799,2	2147,0	2112,5	***2257,6***	2115,3	2137,7	***2293,4***	2280,2	3884,2	3839,0	3752,8	***4161,4***	2017,7	2538,1	2713,1	***2729,5***
Moyenne	1817,0	1950,8	1904,1	***2242,5***	1362,4	1434,7	1461,1	***1492,9***	1603,3	1681,1	1771,2	***1857,0***	2743,2	2775,9	2792,6	***3038,5***	1225,1	1477,0	1509,1	***1997,0***
Médiane	1837,3	1893,8	1950,0	***2327,8***	1359,8	1412,6	1462,0	***1468,7***	1605,5	1681,5	1801,1	***1856,6***	2769,9	2724,6	2795,8	***2920,9***	1222,5	1431,1	1478,0	***1979,8***
Tps. Exéc.	***703,6***	783,1	885,2	1670,9	***487,7***	542,3	614,8	1168,8	***295,5***	330,8	375,1	745,8	***743,7***	824,7	915,6	1718,2	***446,4***	482,3	523,2	915,3
Total Esp.	705	690	719	***649***	646	644	648	***555***	518	510	491	***435***	***733***	738	747	***733***	620	607	577	***499***
Instance	BB11008				BB11033															
Dimension	1372				1482															
Valeur de Γ	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000												
Min	-4,0	37,3	9,6	***69,0***	1650,6	***1694,3***	1367,2	1504,4												
Max	218,9	329,2	***352,4***	325,6	2511,8	***2698,9***	2676,6	2669,8												
Moyenne	112,9	168,8	***189,1***	180,6	2056,5	***2168,7***	2131,9	2157,8												
Médiane	110,7	167,6	***187,6***	164,4	2034,8	2144,7	2142,1	***2157,4***												
Tps. Exéc.	***65,8***	73,5	88,4	200,1	***336,7***	367,4	415,1	663,6												
Total Esp.	169	154	137	***116***	538	530	515	***478***												

Suivant les résultats du Tableau 6-3 et du Tableau 6-4, il est possible de remarquer que l'utilisation de la valeur $\Gamma = 1\,000$ produit les résultats les plus rapides pour 100% des 27 instances. Ceci concorde avec le fait que moins d'efforts de calculs sont demandés pour produire une plus petite quantité d'opérations de magnétisme. Dans le même ordre d'idée, l'utilisation de la valeur $\Gamma = 20\,000$ produit les résultats les plus lents.

Pour analyser les autres informations statistiques, le Tableau 6-5 est construit de manière à présenter un résumé du Tableau 6-3 et du Tableau 6-4. Les résultats sont divisés selon la taille des instances : plus petite que 500, entre 500 et 1 000, de même qu'entre 1 000 et 1 500. Les informations affichées identifient le nombre de fois ou une valeur attribuée à Γ a permis d'obtenir la meilleure statistique de *minimum*, *maximum*, *moyenne*, *médiane* et de *nombre total moyen d'espacements*. La dernière ligne du tableau indique le nombre total de fois où une valeur pour Γ a obtenu la meilleure donnée statistique.

Tableau 6-5 : Sommaire des résultats de la résolution du MSA avec la configuration du nouveau processus *Magnet* selon la taille des instances.

Dimension	< 500				[500, 1 000 [				[1 000, 1 500 [			
Nb. Instances	11				8				8			
Valeur de *Γ*	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000	1 000	5 000	10 000	20 000
Min	2	2	3	4		2	2	4	1	1	2	4
Max	1	5	1	4	2		2	4		1	2	5
Moyenne		2	1	8		1		7		1	1	6
Médiane	1	1	3	6			1	7			1	7
Total Esp.		4	1	7		1		7	1			8
TOTAL	4	14	9	29	2	4	5	29	2	3	6	30

Selon ce dernier tableau, l'utilisation d'une haute valeur de Γ entraîne en général de meilleurs résultats. En effet, pour toutes les données statistiques hormis le temps d'exécution, l'utilisation de Γ défini à 20 000 obtient 44,4% (12/27) du plus grand nombre de meilleurs résultats sur le minimum et 48,1% (13/27) sur le maximum. Il a également obtenu

77,8% (27/27) des meilleurs résultats pour la meilleure moyenne, 74,1% (20/27) pour la meilleure médiane (20/27), de même que 81,5% (22/27) pour le nombre total moyen d'espacements dans les solutions résultantes. En somme avec les totaux pour les 11 instances comportant une dimension inférieure à 500, les 8 instances avec une dimension entre 500 et 1 000 et les 8 instances comportant une dimension entre 1 000 et 1 500, l'utilisation de Γ défini à 20 000 obtient respectivement 52,7% (29/55), 72,5% (29/40) et 75,0% (30/40) des meilleures données statistiques. En comparaison, la deuxième meilleure valeur de Γ correspond à 5 000. Celle-ci détient 25,5% (14/55), 12,5% (5/40) et 15,0% (6/40) des meilleures données statistiques avec les mêmes regroupements d'instances.

En général, bien que l'utilisation de la valeur 20 000 assignée à Γ propose les résultats les plus pertinents pour le magnétisme statique, il ne s'agit pas d'une dominance sur les autres valeurs utilisées. L'utilisation de la valeur 1 000 propose des temps d'exécution bien plus rapides et l'utilisation de la valeur 5 000 détient tout de même de meilleures performances sur une partie des instances. Aucun lien ne peut cependant être tiré concernant le nombre de séquences à résoudre, leur longueur ou encore la différence entre la plus petite et la plus longue séquence de l'instance. Pour cette raison, ces informations ont été omises de cette analyse.

6.4.2.2. MAGNÉSTISME DYNAMIQUE

Une valeur dynamique pour le paramètre de Γ est proposée dans cette section. Cette valeur est conçue de manière à s'ajuster automatiquement selon le comportement de l'essaim. Utiliser une valeur dynamique pour Γ permettrait de bénéficier des avantages provenant des différentes valeurs statiques de la section précédente. Pour signifier la fluctuation des valeurs attribuées avec le magnétisme dynamique, cette technique d'attribution est notée $\Gamma = \mathrm{X}$.

Pour contribuer au facteur social et collectif de ce processus, la fluctuation dynamique proposée pour Γ dans cette section implique l'utilisation du paramètre w modulant la confiance en soi d'une particule. Tel que spécifié à la Section 3.2.2, l'approche utilisée pour l'attribution d'une valeur à w utilise l'Équation 3-5 proposée par Ratnaweera et al. (2004). En effet, la valeur de w s'adapte pour débuter avec une grande valeur afin de favoriser l'exploration de l'espace de recherche, et termine avec une petite valeur pour favoriser l'exploitation de ce même espace. Dans le cas du paramètre Γ, l'inverse de w est utilisé afin de débuter la résolution avec une petite valeur de Γ, réduisant ainsi le nombre maximal d'opérations de magnétisme en début d'exécution. À l'opposé, ce paramètre est défini avec une plus grande valeur vers la fin de la résolution afin d'augmenter le nombre maximal de ces opérations.

L'affectation de la valeur à Γ suit l'Équation 6-5. Le but recherché est que le magnétisme termine avec 20 000 opérations, un nombre représentant la valeur statique ayant obtenu les meilleurs résultats généraux à la section précédente. Reprenant les paramètres idéaux pour le poids d'inertie décrits à la Section 3.2.2, l'algorithme débute avec un w_0 défini à 0,9 et termine son exécution avec un w_1 défini à 0,4. Suivant l'utilisation de $1 - w$ pour l'attribution de la valeur à Γ, l'Équation 6-5 calcule la valeur nécessaire pour que 20 000 soit la valeur affectée à Γ à la toute dernière itération de la mise à jour des sous-essaims du HCoCLPSO-Magnet. La valeur initiale pour Γ est donc de 3 333.

$$\Gamma = (1 - w) \cdot (20\,000 \div (1 - w_1)) \tag{6-5}$$

Les résultats obtenus pour le magnétisme dynamique sont divisés en deux tableaux. Le Tableau 6-6 présente les résultats de la comparaison entre le magnétisme dynamique et le magnétisme statique utilisant la valeur de Γ définie à 20 000 pour les instances comportant une dimension inférieure à 900. Le Tableau 6-7 présente l'équivalent de ces informations pour les instances comportant une dimension entre 900 et 1 500. Les données en gras

italique correspondent au meilleur résultat obtenu entre les deux techniques d'attribution de valeurs pour Γ selon l'instance et les statistiques de *minimum*, *maximum*, *moyenne*, *médiane*, *temps d'exécution* et de *nombre total moyen d'espacement*s. Une nouvelle donnée statistique est ajoutée aux tableaux, laquelle correspond au pourcentage de différence sur chacune des données statistiques entre les deux techniques d'attribution. Une valeur positive signifie que la donnée statistique pour le magnétisme dynamique est plus élevée que la donnée statistique pour le magnétisme statique. Une valeur négative signifie l'inverse.

Tableau 6-6 : Résultats de la résolution du MSA avec la configuration du Magnet utilisant la valeur dynamique de Γ sur les instances comportant une dimension inférieure à 900

Instance	BB11001			BB11025			BB11021			BB11013			BB11029			BB11017		
Dimension	91			161			195			247			252			253		
Valeur de Γ	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.
Min	***78,1***	73,3	-6,1%	-78,3	***-56,8***	-27,5%	-3,0	***47,2***	-1673,3%	***22,2***	-4,8	-121,6%	***48,3***	30,6	-36,6%	126,8	***191,4***	50,9%
Max	***422,3***	399,9	-5,3%	90,3	***122,4***	35,5%	307,6	***339,7***	10,4%	***244,4***	234,1	-4,2%	***334,0***	242,8	-27,3%	***693,3***	645,2	-6,9%
Moyenne	***290,8***	266,1	-8,5%	20,5	***38,2***	86,0%	172,4	***202,2***	17,2%	***128,4***	122,4	-4,7%	***159,1***	144,0	-9,5%	430,7	***433,7***	0,7%
Médiane	***310,9***	276,8	-11,0%	25,4	***42,2***	66,1%	166,4	***206,8***	24,3%	***128,0***	125,0	-2,3%	***151,0***	145,1	-3,9%	***433,6***	423,1	-2,4%
Tps. Exéc.	52,3	***38,3***	-26,9%	51,1	***41,2***	-19,3%	70,5	***56,3***	-20,1%	69,1	***56,3***	-18,5%	63,9	***50,8***	-20,5%	146,2	***112,1***	-23,3%
Total Esp.	***22***	28	27,3%	***31***	43	38,7%	54	***50***	-7,4%	56	***52***	-7,1%	***47***	59	25,5%	83	***79***	-4,8%
Instance	BB11015			BB11035			BB11010			BB11011			BB11012			BB11004		
Dimension	334			338			394			445			448			504		
Valeur de Γ	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.
Min	264,0	***278,6***	5,5%	202,3	***250,5***	23,8%	***-253,8***	-271,3	6,9%	42,9	***55,0***	28,2%	***526,9***	375,9	-28,7%	***-65,7***	-114,5	74,3%
Max	1072,3	***1150,5***	7,3%	536,4	***544,3***	1,5%	***447,1***	408,4	-8,7%	***520,2***	465,3	-10,6%	***1547,7***	1199,4	-22,5%	***352,4***	333,9	-5,2%
Moyenne	673,6	***709,7***	5,4%	382,5	***387,7***	1,3%	***39,2***	34,7	-11,3%	265,1	***265,4***	0,1%	***864,5***	806,5	-6,7%	137,4	***150,2***	9,4%
Médiane	696,3	***709,5***	1,9%	389,1	***389,3***	0,1%	21,5	***35,7***	66,0%	***265,8***	263,2	-1,0%	***842,3***	812,3	-3,6%	149,4	***157,6***	5,5%
Tps. Exéc.	187,9	***142,7***	-24,0%	94,4	***75,7***	-19,8%	303,1	***230,6***	-23,9%	195,8	***156,2***	-20,2%	242,7	***179,4***	-26,1%	282,3	***218,1***	-22,7%
Total Esp.	108	***98***	-9,3%	87	***82***	-5,7%	***143***	162	13,3%	***152***	163	7,2%	***107***	110	2,8%	179	***176***	-1,7%
Instance	BB11022			BB11020			BB11003			BB11024			BB11009			BB11006		
Dimension	512			583			596			604			752			754		
Valeur de Γ	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.
Min	-8,8	***10,7***	-221,6%	-82,3	***147,1***	-278,7%	***46,5***	-15,3	-132,9%	-76,8	***-26,5***	-65,5%	***28,2***	10,1	-64,2%	-141,9	***-134,5***	-5,2%
Max	***225,1***	188,4	-16,3%	***1292,7***	1154,9	-10,7%	421,8	***492,9***	16,9%	***430,3***	353,7	-17,8%	***265,6***	249,2	-6,2%	***857,5***	754,3	-12,0%
Moyenne	***107,4***	106,4	-0,9%	***654,1***	601,0	-8,1%	***248,0***	243,4	-1,8%	***184,1***	171,0	-7,1%	***160,6***	132,7	-17,4%	***378,6***	331,0	-12,6%
Médiane	***114,6***	100,6	-12,2%	***684,7***	585,3	-14,5%	233,9	***261,3***	11,7%	174,3	***174,4***	0,1%	***156,2***	134,9	-13,6%	***394,5***	352,1	-10,7%
Tps. Exéc.	86,3	***65,1***	-24,6%	680,6	***596,0***	-12,4%	325,7	***250,1***	-23,2%	286,8	***222,8***	-22,3%	152,5	***114,3***	-25,1%	650,3	***565,0***	-13,1%
Total Esp.	***57***	***57***	0,0%	***318***	328	3,1%	***185***	195	5,4%	***161***	204	26,7%	96	***93***	-3,1%	***369***	382	3,5%

Tableau 6-7 : Résultats de la résolution du MSA avec la configuration du Magnet utilisant la valeur dynamique de Γ sur les instances comportant une dimension entre 900 et 1 500

Instance	BB11028			BB11002			BB11007			BB11036			BB11023			BB11019		
Dimension	942			1064			1167			1224			1247			1266		
Valeur de Γ	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.
Min	598,0	***615,2***	2,9%	***597,9***	560,3	-6,3%	***1038,1***	884,1	-14,8%	***966,1***	903,2	-6,5%	1201,3	***1253,2***	4,3%	***2296,8***	1739,7	-24,3%
Max	***1698,0***	1627,7	-4,1%	1131,6	***1180,8***	4,3%	3245,1	***3542,9***	9,2%	2257,6	***2276,9***	0,9%	2280,2	***2505,6***	9,9%	***4161,4***	3717,0	-10,7%
Moyenne	1108,9	***1112,5***	0,3%	***847,4***	839,8	-0,9%	***2242,5***	2067,9	-7,8%	1492,9	***1514,4***	1,4%	***1857,0***	1786,9	-3,8%	***3038,5***	2742,8	-9,7%
Médiane	1109,2	***1116,2***	0,6%	***845,2***	835,4	-1,2%	***2327,8***	1998,9	-14,1%	1468,7	***1496,0***	1,9%	***1856,6***	1785,1	-3,9%	***2920,9***	2771,6	-5,1%
Tps. Exéc.	648,3	***575,0***	-11,3%	273,2	***235,3***	-13,9%	1670,9	***1485,7***	-11,1%	1168,8	***1032,0***	-11,7%	745,8	***643,1***	-13,8%	1718,2	***1552,6***	-9,6%
Total Esp.	***365***	371	1,6%	***219***	237	8,2%	***649***	661	1,8%	***555***	577	4,0%	435	***432***	-0,7%	733	***718***	-2,0%

Instance	BB11014			BB11008			BB11033		
Dimension	1323			1372			1482		
Valeur de Γ	20 000	X	% Diff.	20 000	X	% Diff.	20 000	X	% Diff.
Min	***1234,6***	985,0	-20,2%	***69,0***	16,9	-75,5%	1504,4	***1540,8***	2,4%
Max	2729,5	***3259,1***	19,4%	325,6	***346,4***	6,4%	2669,8	***2906,4***	8,9%
Moyenne	***1997,0***	1719,9	-13,9%	180,6	***189,5***	4,9%	2157,8	***2160,6***	0,1%
Médiane	***1979,8***	1633,2	-17,5%	164,4	***195,0***	18,6%	***2157,4***	2142,9	-0,7%
Tps. Exéc.	915,3	***896,1***	-2,1%	200,1	***185,0***	-7,6%	663,6	***618,1***	-6,9%
Total Esp.	***499***	582	16,6%	***116***	137	18,1%	***478***	512	7,1%

Concernant les temps d'exécution, le magnétisme dynamique propose de meilleurs temps que le magnétisme statique avec la valeur de Γ définit à 20 000, et ce, pour toutes les 27 instances de problèmes. Ces résultats sont conséquents avec l'implémentation du nouveau processus Magnet. En effet, le nombre d'opérations de magnétisme varie selon l'exécution et obtient, au maximum, 20 000 opérations. Il est donc logique que le magnétisme dynamique soit toujours plus rapide que le magnétisme statique.

Le Tableau 6-8 présente un résumé du nombre de fois où une technique d'attribution a obtenu les meilleures données statistiques selon la taille des instances. La dernière ligne du tableau indique le nombre total de fois où une technique d'attribution a obtenu la meilleure donnée statistique. Les résultats sont divisés selon la taille des instances : inférieure à 500, entre 500 et 1 000, de même qu'entre 1 000 et 1 500.

Tableau 6-8 : Sommaire des résultats de la résolution du MSA entre le magnétisme statique et le magnétisme dynamique du nouveau processus *Magnet* selon la taille des instances.

Dimension	< 500		[500, 1 000 [		[1 000, 1 500 [	
Nb. Instances	11		8		8	
Valeur de Γ	20 000	X	20 000	X	20 000	X
Min	5	6	3	5	6	2
Max	7	4	7	1	1	7
Moyenne	5	6	6	2	5	3
Médiane	6	5	4	4	6	2
Total Esp.	6	5	6	2	6	2
TOTAL	29	26	26	14	24	16

Suivant les résultats des trois tableaux précédents, il est difficile de distinguer quelle technique d'attribution fonctionne mieux que l'autre. En effet, le magnétisme statique obtient les meilleures données statistiques pour 52,7% (29/55) des instances comportant une taille inférieure à 500, 65% (26/40) des instances de taille entre 500 et 1 000 et 60,0% (24/40) des instances de taille entre 1 000 et 1 500. En contrepartie, le magnétisme dynamique obtient

47,3% (26/55), 35,0% (14/40) et 40,0% (16/40) des meilleures données statistiques pour les mêmes divisions d'instances. Ceci signifie que les deux techniques d'attribution fonctionnent bien et qu'il est difficile, somme toute, de les départager considérant la non-dominance des résultats. Tout comme pour les tests sur le paramétrage avec l'attribution statique, aucune caractéristique visuelle sur les instances ne permet d'établir de corrélation avec les résultats obtenus dans cette section. Ces informations ont donc été omises dans les tableaux précédents.

En observant plus attentivement les résultats au Tableau 6-6 et au Tableau 6-7, plusieurs instances comportent des résultats très similaires pour les deux techniques d'attribution. Par exemple, les instances BB11002, BB11011, BB11028, BB1033 et BB11035 comportent toutes une différence de moins de 1,5% concernant les moyennes et les médianes pour les 50 exécutions. Une différence si peu significative ne permet pas de déterminer qu'une technique d'attribution est meilleure qu'une autre. Le Tableau 6-9 met en évidence le pourcentage de différence moyen pour les statistiques présentées au Tableau 6-6 et au Tableau 6-7.

Tableau 6-9 : Sommaire des pourcentages de différence pour la résolution des 27 instances de MSA selon les résultats provenant du Tableau 6-6 et du Tableau 6-7.

% Diff. Moyen	
Min	-96,7%
Max	-1,4%
Moyenne	0,1%
Médiane	2,9%
Tps. Exéc.	-17,6%
Total Esp.	6,3%

Tel qu'il est démontré dans le tableau précédent, le pourcentage de différence entre la moyenne générale de l'attribution dynamique comparée à celle de l'attribution statique est de 0,1%. Le pourcentage de différence concernant la médiane générale de ces mêmes

attributions correspond quant à elle à 2,9%. Ceci signifie que le magnétisme dynamique propose des moyennes et des médianes légèrement meilleures que celles obtenues avec le magnétisme statique. En revanche, les pourcentages de divergence de minimums (-96,7%), de maximums (-1,4%) et du nombre total moyen d'espacements (6,3%) favorisent l'utilisation du magnétisme statique. Une différence aussi grande pour la divergence de minimums implique que l'exploration de l'espace de recherche avec le magnétisme dynamique est moins efficace en début d'algorithme, ce qui est cohérent avec le nombre d'opérations de magnétisme utilisé pour cette technique.

En général, aucune des deux techniques d'attribution ne domine l'autre. Au contraire, elles semblent toutes les deux être complémentaires pour la résolution de l'ensemble des instances. Cependant, l'utilisation du magnétisme dynamique nécessite moins de temps pour résoudre les problèmes. Considérant les résultats présentés dans cette section, la section suivante compare les deux techniques d'attribution proposées dans cette section avec deux outils utilisés en bio-informatique.

6.4.3. COMPARAISON DU HCOCLPSO-MAGNET AVEC DEUX APPROCHES RECONNUES

Cette section vise à analyser et à comparer les performances de la nouvelle hybridation HCoCLPSO-Magnet avec les résultats provenant de la Section 6.3, de même qu'avec des outils commerciaux utilisés en bio-informatique. Pour ce faire, deux outils pour la résolution du MSA sont sélectionnés selon deux des approches mentionnées à la Section 6.2.4. Le premier outil, nommé MAFFT (Katoh et al., 2002), utilise une approche itérative et le deuxième outil, nommé CLUSTAL (Hung et al., 2015; Larkin et al., 2007), utilise une approche progressive.

Les paramètres du MSA utilisés pour la résolution des instances sont mentionnés à la Section 6.2.5. Pour obtenir les résultats de la comparaison avec MAFFT et CLUSTAL, les instances ont été exécutées directement sur le site Internet respectif de chaque outil (MAFFT : https://mafft.cbrc.jp/alignment/server/, CLUSTAL : https://www.genome.jp/tools-bin/clustalw). La version de CLUSTAL utilisée est CLUSTALW considérant qu'il s'agit de celle dont il est possible de paramétrer les pénalités d'ouverture et d'extension des espacements, de même que la matrice de substitution utilisée. Les fichiers de résultats contenant les instances résolues ont été évalués avec la fonction objectif du HCoCLPSO-Magnet pour obtenir une valeur comparable des solutions résultantes. La base des solutions est ainsi la même pour chaque méthode.

Le Tableau 6-10 présente les statistiques de la comparaison entre le meilleure procédé de discrétisation de la Section 6.3 (LT-Index), les deux techniques d'attribution de magnétisme du HCoCLPSO-Magnet et les deux outils MAFFT et CLUSTALW. Considérant que MAFFT et CLUSTALW ne fournissent qu'une seule solution de manière déterministe et identique pour chaque exécution, seulement les meilleures exécutions de LT-Index, de HCoCLPSO-Magnet pour le magnétisme statique et pour le magnétisme dynamique sont utilisées. Les *moyennes*, les *minimums* et les *médianes* ont donc été filtrés. Les temps d'exécution ne peuvent pas non plus être comparés considérant que les systèmes ayant exécuté les résolutions ne sont pas les mêmes.

Tableau 6-10 : Résultats de la comparaison entre la discrétisation LT-Index, l'hybride HCoCLPSO-Magnet (statique et dynamique), MAFFT et CLUSTALW

Instance	Dimension	LT-Index	Γ = 20 000	Γ = X	MAFFT	CLUSTALW
BB11001	91	296,2	***422,3***	399,9	351	364,5
BB11025	161	78,9	90,3	***122,4***	-85,9	-164,2
BB11021	195	233,5	307,6	***339,7***	316,7	-102,3
BB11013	247	183,8	***244,4***	234,1	-173	-260,5
BB11029	252	159,7	***334,0***	242,8	90,4	50,7
BB11017	253	329,8	***693,3***	345,2	637,7	290,5
BB11015	334	707,5	1072,3	***1150,5***	542	269,2
BB11035	338	410,7	536,4	***544,3***	368,1	207,6
BB11010	394	42,6	***447,1***	408,4	110,8	-346,4
BB11011	445	329,1	***520,2***	465,3	-667,1	-877,3
BB11012	448	676,2	1547,7	1199,4	***1690,6***	1532,1
BB11004	504	72	***352,4***	333,9	88,4	-686,7
BB11022	512	135	***225,1***	188,4	-170	-59
BB11020	583	1043,5	***1292,7***	1154,9	1266,2	-190,7
BB11003	596	157,6	421,8	492,9	***641,5***	617,8
BB11024	604	171,6	***430,3***	353,7	133,7	-779,4
BB11009	752	156,9	265,6	249,2	***334,6***	39,1
BB11006	754	661,1	***857,5***	754,3	-1044,7	-2025,4
BB11028	942	1383,7	***1698,0***	1627,7	284	-457,6
BB11002	1064	896,6	1131,6	***1180,8***	707,2	-207,6
BB11007	1167	3060,4	3245,1	***3542,9***	2013,5	1970
BB11036	1224	1890,9	2257,6	***2276,9***	-452	-841,3
BB11023	1247	2101,6	2280,2	***2505,6***	-367,4	-444
BB11019	1266	3578	***4161,4***	3717,0	1567,3	1250,5
BB11014	1323	2157,4	2729,5	3259,1	***3287,2***	3191,6
BB11008	1372	155,1	325,6	***346,4***	248,2	27,5
BB11033	1482	2607,6	2669,8	***2906,4***	-206,4	-416,8
TOTAL		0	13	10	4	0

Le tableau précédent démontre la capacité de l'hybride HCoCLPSO-Magnet à rivaliser avec les outils reconnus du marché. Il ne démontre pas une dominance complète sur MAFFT et CLUSTALW, mais il arrive tout de même à proposer de meilleures solutions selon l'évaluation par la fonction objectif pour 85,2% (23/27) des instances de MSA. MAFFT obtient

14,8% (4/27) des meilleures solutions et CLUSTALW n'en obtient aucune. Il n'est pas possible de tirer des conclusions pour cette comparaison en fonction des caractéristiques des instances telles que le nombre de séquences à aligner ou encore leurs longueurs. Bien que le procédé de discrétisation LT-Index parvient à offrir des solutions meilleures que MAFFT et CLUSTALW pour 59,3% (16/27) des instances, il est dominé entièrement par les deux techniques d'attribution de l'hybride HCoCLPSO-Magnet. Ceci démontre l'efficacité de l'hybridation utilisée, avec la modélisation de solutions en anneau, comparativement à une méthode ne comportant pas d'hybridation.

La Figure 6-17 illustre les résultats obtenus au Tableau 6-10. Elle met en évidence les différences entre les résultats obtenus pas la fonction objectif dans le contexte des expérimentations. Les traits pleins illustrent les propositions de ce chapitre (les deux techniques d'attribution de magnétisme du HCoCLPSO-Magnet) et les traits pointillés illustrent les deux outils (MAFFT et CLUSTALW). Pour simplifier de comparaison, les résultats provenant du procédé de discrétisation LT-Index ne sont pas représentés considérant qu'il est dominé par les deux techniques d'attribution de magnétisme.

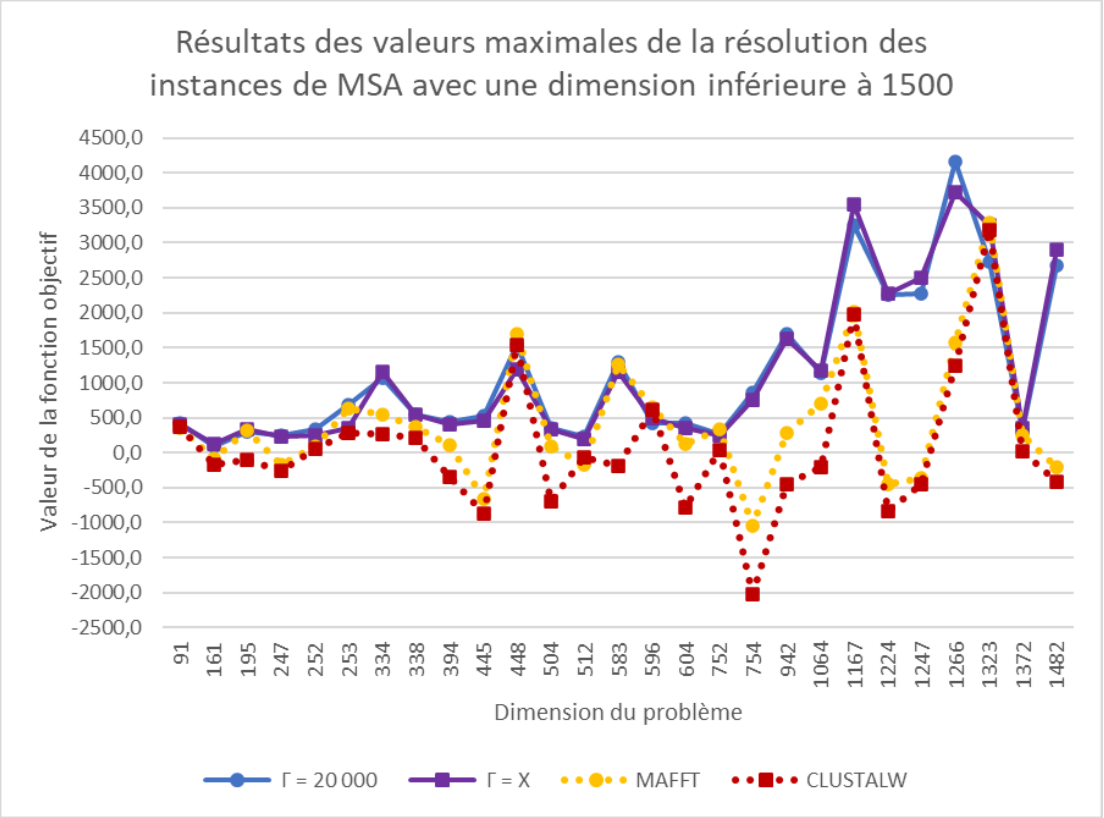

Figure 6-17 : Représentation visuelle des résultats provenant du Tableau 6-10.
© Hugo Deschênes

Tel qu'illustré à la figure précédente, les différences dans les résultats provenant de l'évaluation par la fonction objectif sont de plus en plus notables lors de la résolution d'instances de grande taille. À partir de la dimension 754, les résultats provenant de l'hybride HCoCLPSO-Magnet sont nettement supérieurs ou très près de ceux provenant des outils MAFFT et CLUSTALW. Également, la Figure 6-17 met en évidence que les résultats proposés par les deux versions du magnétisme proposent des résultats similaires.

Le Tableau 6-10 et la Figure 6-17 permettent d'établir que l'hybride HCoCLPSO-Magnet rivalise avec les outils utilisés en bio-informatique en termes de qualité des valeurs obtenues avec la fonction objectif. Cependant, le nombre d'ouvertures d'espacements n'a pas été démontré et analysé dans cette section. Effectivement, bien que les résultats

obtenus avec l'hybride HCoCLPSO-Magnet soient excellents, cet hybride a beaucoup de difficulté à produire une solution comportant un bas nombre d'ouvertures.

Un exemple d'alignement résolu est utilisé afin d'analyser adéquatement la comparaison des résultats. La Figure 6-18 illustre la meilleure solution provenant de la résolution de la plus petite instance disponible (BB11001) avec les quatre méthodes utilisées dans cette section, soient CLUSTALW, MAFFT et les deux techniques d'attribution du HCoCLPSO-Magnet ($\Gamma = 20\,000$ et $\Gamma = X$). Le numéro des index est indiqué à la première ligne pour faciliter la lecture de l'alignement. Pour le reste de cette section, les résultats de la discrétisation LT-Index ne sont plus mentionnés considérant que cette méthode de résolution est dominée par l'hybride HCoCLPSO-Magnet.

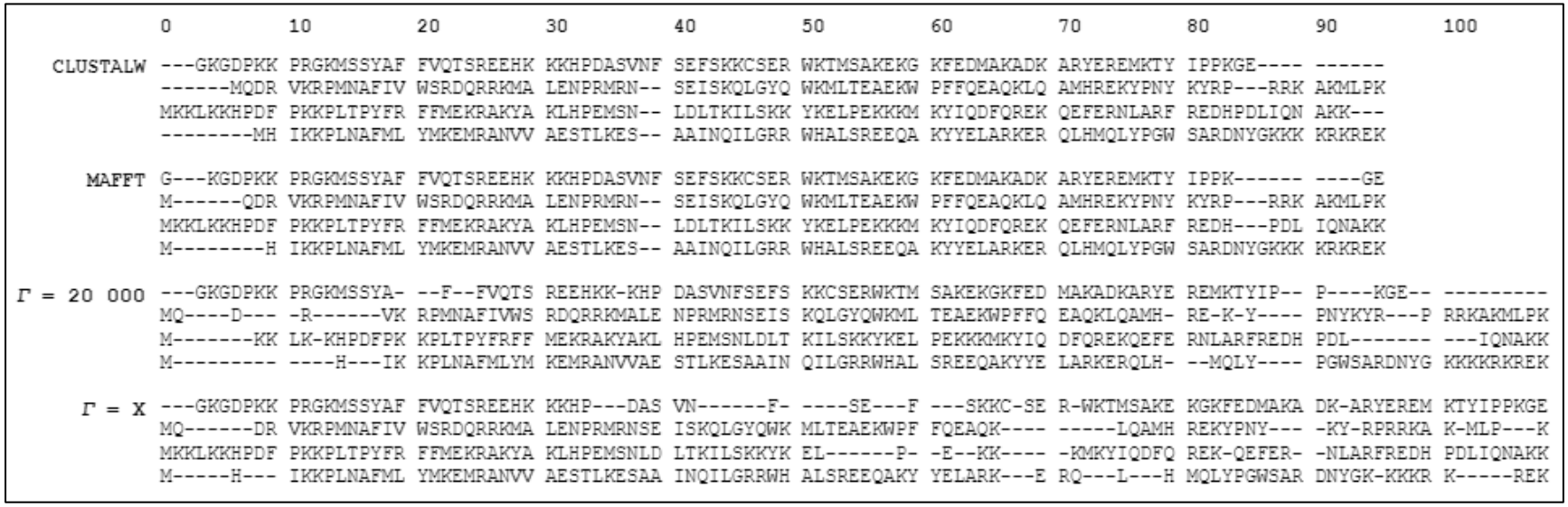

Figure 6-18 : Résultats visuel de la solution provenant de la résolution de l'instance BB11022 pour l'hybride HCoCLPSO-Magnet, MAFFT et CLUSTALW.

Il est important de noter que chaque méthode de résolution produit des résultats comportant des caractéristiques visuelles similaires, et ce, peu importe l'instance utilisée. BB11001 est sélectionnée pour la figure précédente à titre de référence pour cette analyse considérant sa plus petite taille. Les conclusions tirées sont transposables aux 26 autres instances résolues pour cette section.

Il est possible d'observer dans cette figure que les instances résolues avec les outils MAFFT et CLUSTALW possèdent un bas nombre d'espacements. Les espacements sont davantage regroupés en comparaison des solutions obtenues avec les deux techniques d'attribution de magnétisme du HCoCLPSO-Magnet ($\Gamma = 20\,000$ et $\Gamma = X$). Également, les résultats provenant du HCoCLPSO-Magnet utilisent un plus grand nombre d'espaces pour produire des résultats en raison de la méthode de résolution utilisée.

Le Tableau 6-11 présente un résumé des caractéristiques visuelles provenant de la Figure 6-18. Les détails de l'évaluation de chaque solution par la fonction objectif y sont inscrits selon la méthode utilisée pour résoudre l'instance. Les variables α, β et SP proviennent de l'Équation 6-1, de l'Équation 6-2 et de l'Équation 6-3 pour le calcul du score de l'alignement multiple de séquences. α représente le score relié à l'alignement des acides aminés (degré de similitude) et β représente le total des pénalités accordées aux espacements dans l'ensemble de l'alignement. SP représente le résultat provenant de l'évaluation par la fonction objectif, soit la soustraction de β à α. Les pénalités pour chaque séquence de l'alignement (Pén.) et le nombre d'ouvertures (Nb. Ouv.) sont présentés pour les quatre séquences (Séq.).

Tableau 6-11 : Caractéristiques des espacements pour la résolution de l'instance BB11001 avec HCoCLPSO-Magnet (statique et dynamique), MAFFT et CLUSTALW.

	BB11001							
	Γ = 20 000		Γ = X		MAFFT		CLUSTALX	
	Pén.	Nb. Ouv.	Pén.	Nb. Ouv.	Pén.	Nb. Ouv.	Pén.	Nb. Ouv.
Séq. 1	-25,7	7	-41,5	9	-11,1	2	0	2
Séq. 2	-41,6	8	-31,8	6	-15,8	3	-10,3	3
Séq. 3	-16,5	3	-31,2	6	-10,3	2	-5,1	2
Séq. 4	-21,9	4	-36,6	7	-10,8	2	-5,1	2
Total β	-105,7	22	-141,1	28	-48,0	9	-20,5	9
SP	422,3		399,9		351,0		364,5	
α	528,0		541,0		399,0		385,0	

© Hugo Deschênes

Le tableau précédent met en évidence deux éléments importants dans le calcul de la fonction objectif d'un alignement multiple de séquences : la maximisation du score d'alignement de caractères similaires et la minimisation des pénalités causées par les espacements. Concernant la première portion de l'évaluation, la résolution par l'hybride HCoCLPSO-Magnet produit des alignements avec des caractères qui se ressemblent davantage avec des scores de 528,0 pour le magnétisme statique (Γ = 20 000) et de 541,0 pour le magnétisme dynamique (Γ = *X*). En contrepartie, MAFFT et CLUSTALW produisent un alignement comportant un score de 399,00 et de 385,0 respectivement, ce qui est nettement inférieur. À propos de la deuxième portion de l'évaluation, MAFFT et CLUSTALW produisent des alignements comportant une plus petite pénalité causée par les espacements avec un score de -48 et de -20,5 respectivement. En revanche, le magnétisme statique et le magnétisme dynamique produisent une pénalité d'espacements de -105,7 et de -141,1 respectivement.

En somme, l'utilisation du HCoCLPSO avec la nouvelle modélisation de solutions en anneau, combinée au nouveau processus d'apprentissage par magnétisme, produit un plus grand nombre de regroupements d'espacements en comparaison à deux outils utilisés en bio-informatique, soient MAFFT et CLUSTALW. Cependant, cela s'avère au final bénéfique

en considérant l'ensemble des éléments de la fonction objectif afin de produire des solutions du MSA favorisant l'alignement des caractères similaires.

6.5. CONCLUSION SUR LA CONTRIBUTION

Ce chapitre représente la combinaison entre les résultats provenant du Chapitre 4 et du Chapitre 5. Il vise à combiner un procédé de discrétisation et une stratégie d'hybridation afin de concevoir un algorithme capable de résoudre efficacement des instances de grande taille provenant du problème MSA.

Après avoir défini le fonctionnement, les enjeux et les techniques pour résoudre le MSA, deux nouvelles discrétisations avec codification des solutions en anneau ont été décrites et expérimentées. Celles-ci sont basées sur la liste de translations d'Anghinolfi et Paolucci (2009) et comparent l'utilisation de variables de décision entières représentant les index d'un espace e (LT-Index) avec l'utilisation de variables de décision réelles représentant la proportion de la longueur de la séquence s (LT-Proportion). La nouvelle modélisation de solution utilisée en anneau joint le début et la fin de l'espace de recherche dans le but d'éviter aux particules de se contraindre à l'intérieur d'un espace. Elle permet également de rendre le PSO plus malléable pour effectuer une hybridation performante. Il a été confirmé que l'utilisation de variables discrètes avec la codification des solutions en anneau propose de meilleures solutions pour l'entièreté des 27 instances de MSA.

Suivant cette conclusion, le nouveau processus d'apprentissage nommé Magnet a été proposé, lequel remplace le processus d'apprentissage compréhensif de l'hybride HCoCLPSO. L'hybride final ainsi formé se nomme HCoCLPSO-Magnet. Ce nouveau processus d'apprentissage reproduit un comportement de magnétisme entre les espacements des séquences provenant d'une instance de MSA. Il utilise également la

codification entière de solutions en anneau pour aider les espacements en début/fin de séquence à se regrouper ensemble. Les expérimentations pour ajuster adéquatement la valeur attribuée au paramètre Γ, lequel sert à définir le nombre maximal d'appels au magnétisme, ont par la suite permis de déterminer deux types de magnétisme : statique et dynamique. Concernant le magnétisme statique, la définition de la valeur 20 000 propose les meilleurs résultats généraux. Le magnétisme dynamique vise quant à lui à modifier la valeur attribuée à Γ à travers les itérations de l'algorithme. Suivant la comparaison entre le magnétisme statique et le magnétisme dynamique, aucune des deux techniques d'attribution pour Γ ne semble dominer l'autre. Le magnétisme statique propose en moyenne de meilleurs espacements, de même que de meilleurs minimums et maximums. En contrepartie, le magnétisme dynamique propose en moyenne un meilleur temps d'exécution, de même que de meilleures moyennes et médianes. Ils sont donc considérés équivalents, voire complémentaires.

Pour finir, l'hybride HCoCLPSO-Magnet a été comparé avec le meilleure procédé de discrétisation (LT-Index), ainsi que deux outils reconnus en bio-informatique selon deux approches différentes : l'approche itérative avec MAFFT et l'approche progressive avec CLUSTALW. Il a été prouvé que l'hybride HCoCLPSO-Magnet domine les résultats proposés par l'utilisation du procédé LT-Index sans hybridation. Il réussit également à présenter les meilleurs résultats selon l'évaluation par la fonction objectif pour 85,2% des 27 instances utilisées comportant une dimension inférieure à 1 500. Bien que MAFFT et CLUSTALW sont en mesure de minimiser les pénalités causées par les espacements, l'hybride HCoCLPSO-Magnet performe mieux pour maximiser le score de l'alignement des colonnes d'acides aminés.

Plus globalement, ce chapitre a permis d'étendre l'applicabilité du PSO par la conception d'un procédé de discrétisation entier avec une nouvelle modélisation de solutions en anneau, jumelé à un nouveau processus d'apprentissage nommé Magnet. Cette

conception de l'hybride HCoCLPSO-Magnet a permis de *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA*.

CHAPITRE 7 – CONCLUSION ET PERSPECTIVES

7.1. RETOUR SUR LA THÈSE ET LES CONTRIBUTIONS

Cette conclusion est construite comme suit. Tout d'abord, un bref retour sur les sujets abordés est présenté. Un accent plus prononcé est mis sur l'atteinte des objectifs de la recherche et les contributions effectuées à la communauté scientifique. Ensuite, les possibilités de contributions supplémentaires sont présentées. En effet, cette thèse aborde des sujets expérimentés dans un contexte précis. Elle permet donc des extensions afin d'élargir leur portée, et ainsi contribuer davantage à l'amélioration des stratégies métaheuristiques pour la résolution de grandes instances de problèmes. Finalement, les perspectives d'améliorations sont abordées, proposant certaines avenues, stratégies et perspectives de recherche pour la poursuite de ce qui est présenté dans cet ouvrage.

La résolution de problèmes combinatoires apporte un lot de difficultés reliées entre autres à l'augmentation de la taille du problème à résoudre. Il y a ainsi une explosion de la quantité de solutions potentielles. La performance de certaines métaheuristiques se détériore rapidement lorsque la dimension d'un problème augmente, entraînant ainsi une difficulté à équilibrer l'exploration et l'exploitation lors du parcours des solutions dans l'espace de recherche. Cette thèse aborde ces problématiques par la conception de stratégies de conception telles que la discrétisation et l'hybridation. Ces stratégies permettent d'adapter un algorithme et d'améliorer les performances des métaheuristiques à base de population lors de la résolution de grandes instances de problèmes. Pour accompagner ces stratégies, le PSO et ses variantes sont utilisés pour la résolution de trois problèmes : le SOP, le RCPSP et le MSA.

Le Chapitre 2 amorce le début de la revue de la littérature pour cette thèse. Il sert à établir les bases et les stratégies utilisées dans la littérature pour résoudre des problèmes avec les métaheuristiques. Plus précisément, ils ont permis de comprendre les enjeux et les développements permettant à une métaheuristique de résoudre une plus grande variété de

problèmes que ceux pour lesquelles cette métaheuristique a été conçue à l'origine. Les deux stratégies détaillées sont la discrétisation et l'hybridation.

Le Chapitre 3 présente le PSO comme métaheuristique performante et polyvalente pour la résolution de problèmes continus. Il y est décrit son fonctionnement, certaines de ses variantes, de même que les stratégies d'hybridation et les procédés de discrétisation utilisant cette métaheuristique. Les objectifs de cette thèse terminent ce chapitre en utilisant le PSO comme stratégie métaheuristique pour améliorer la résolution de grandes instances de problèmes avec la proposition de nouvelles stratégies d'hybridations et de nouveaux procédés de discrétisation.

Les contributions effectuées aux Chapitres 4, 5 et 6 ont permis de valider l'objectif principal, consistant à *Contribuer aux stratégies métaheuristiques à base de population pour résoudre efficacement des problèmes de grande taille*. Cet objectif est atteint par la rencontre de trois objectifs secondaires :

- *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature* ;
- *Adapter et comparer trois méthodes de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support* ; et
- *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA.*

Une première contribution est présentée au Chapitre 4. Celle-ci consiste à concevoir et comparer des modèles hybrides utilisant des variantes de PSO, soient le BPSO, le CLPSO et le CoLPSO. Trois modèles hybrides sont ainsi proposés : le HCLBPSO-Half, le HBPSO+CL et le HCoCLPSO. Ces modèles sont mis à l'épreuve à l'aide de fonctions continues classiques et des problèmes provenant de CEC'15.

Le premier objectif secondaire, qui est de *Proposer un schéma d'hybridation original combinant efficacement différentes variantes du PSO existantes dans la littérature*, est atteint grâce aux modèles hybrides conçus dans ce chapitre. Les expérimentations effectuées permettent de déterminer que parmi ces modèles, l'hybride HCoCLPSO est l'hybridation offrant les meilleures performances pour la résolution de fonctions continues classiques et des fonctions provenant de l'ensemble CEC'15. Une publication dans une revue scientifique démontre les bénéfices de l'hybridation entre différentes variantes de PSO. La référence est la suivante :

> Deschenes, H., & Gagne, C. (2020). *Solving high dimensional multimodal continuous optimisation problems using hybridisation between particle swarm optimisation variants*. International Journal of Metaheuristics, 7(3), 239-264.
> DOI : 10.1504/IJMHEUR.2020.107390.

Au Chapitre 5, trois procédés de discrétisations sont proposés et adaptés : par priorité d'ordonnancement, par liste de translations et par liste de permutations. Ces procédés provenant de la littérature sont difficilement comparables considérant qu'ils sont expérimentés sur des problèmes différents, dans des contextes différents. Ils sont adaptés en fonction de deux problèmes de support présentant différents niveaux de contraintes : le SOP et le RCPSP. Ces adaptations rendent ainsi possible la comparaison des procédés de discrétisation sur une base commune. Elles permettent également de déterminer lequel offre la performance moyenne la plus robuste.

Le deuxième objectif secondaire, qui est d'*Adapter et comparer trois méthodes de discrétisation provenant de la littérature sur une base commune pour la résolution de divers problèmes de support*, est atteint avec les contributions de ce chapitre. Le procédé de discrétisation ayant permis d'obtenir les meilleurs résultats pour la résolution du SOP et du RCPSP est celui par liste de translations. Il s'agit de celui ayant permis d'obtenir la performance moyenne la plus robuste pour la résolution de problèmes combinatoires à

complexité variable, et ce, avec des instances de différentes tailles. Un acte de conférence est publié et démontre l'apport de la discrétisation avec le PSO. La référence est la suivante :

> Deschênes H., Gagné C., Discrétisation de l'optimisation par essaims particulaires sur une base commune, 11th International Conference on Modeling, Optimization & Simulation (MOSIM 2016), Montréal, Canada, 22-24 August 2016.

Le Chapitre 6 complète cette thèse par la proposition de nouvelles stratégies utilisant les contributions du Chapitre 4 et du Chapitre 5. La conception d'un nouvel hybride de PSO, nommé HCoCLPSO-Magnet, a été proposée. Cet hybride amène deux nouvelles idées pour la conception de métaheuristiques : la discrétisation du CoLPSO pour la résolution du MSA avec une nouvelle modélisation de solutions en anneau et le processus d'apprentissage Magnet. Ils sont testés pour la résolution des instances de MSA provenant de l'ensemble RV11, et comparés avec deux outils utilisés en bio-informatique : MAFFT et CLUSTAL.

Le troisième objectif secondaire, qui est de *Contribuer à l'expansion du PSO par la définition d'un nouveau comportement performant pour la résolution de grandes instances du problème du MSA*, est atteint avec les nouvelles stratégies proposées dans ce chapitre. En effet, il est démontré que la discrétisation par listes de translations utilisant des variables de décision entières, combinée avec la codification des solutions en anneau, permet d'aider les espaces d'un alignement multiple de séquence à se regroupe en début et en fin de séquence. Il aide le PSO à conserver son inertie et enlève les bornes du problème en créant un espace de recherche non limité. Cette amélioration rend le PSO plus malléable et permet de concevoir de nouvelles stratégies de conception. Pour suivre cette idée, le nouveau processus d'apprentissage Magnet utilise cette codification et permet d'offrir de bonnes performances pour la résolution du MSA en adaptant le HCoCLPSO. Ce processus vise à rassembler les espacements ensemble, diminuant ainsi les pénalités accorées aux ouvertures d'espacements. Le paramétrage du nouveau processus Magnet démontre que l'attribution statique à 20 000 pour le nombre maximal d'opérations de magnétisme Γ et

l'attribution dynamique X sont complémentaires pour obtenir les meilleures solutions. Bien que l'hybride HCoCLPSO-Magnet détient davantage de meilleures solutions selon l'évaluation par la fonction objectif que celles provenant de MAFFT et CLUSAL, il a plus de difficulté à regrouper les espacements ensemble pour diminuer la pénalité reliée à ces espacements. Cependant, en comparaison avec ces deux outils, il excelle dans la constitution d'une solution présentant une fonction objectif de qualité pour l'alignement des caractères similaires. Deux actes de conférences sont soumis pour ces contributions. Un article étendu est également en préparation pour une revue scientifique. Les référence de cette contribution sont les suivantes :

Deschênes, H., Gagné, C. (2022). *Solving the Multiple Sequence Alignment Problem using a New Ring Discretization Technique*.

Deschênes, H., Gagné, C. (2022). *Using Magnetism with PSO Hybrid Variants to Solve the Multiple Sequence Alignment Problem*.

7.2. PORTÉE DES CONTRIBUTIONS

La majorité des nouvelles métaheuristiques sont à base de population et sont conçues majoritairement pour la résolution de problèmes continus (Fister Jr et al., 2013; Hussain et al., 2019). Les nouvelles stratégies de conception proposées dans cette thèse sont transposables à ces métaheuristiques continues à base de population, de même que pour la résolution d'une plus grande variété de problèmes. Tel que mentionné à la Section 2.2.3, la clé de la performance d'une métaheuristique réside dans la manière qu'elle est adaptée pour la résolution d'un problème précis.

Tout d'abord, les procédés de discrétisation du Chapitre 4 sont naturellement transposables à plusieurs métaheuristiques continues d'intelligence en essaims. Il est donc facile d'imaginer ces procédés utilisés sur une autre métaheuristique, comme par exemple, sur l'évolution différentielle (Storn & Price, 1997). Cette métaheuristique utilise entre autres

une série d'opérateurs arithmétiques afin de mettre à jour les solutions. La redéfinition de ces opérateurs est un concept de base pour la discrétisation.

Ensuite, la stratégie d'hybridation sélectionnée au Chapitre 5 consiste à ajouter le processus d'apprentissage compréhensif provenant du CLPSO au début de la mise à jour des sous-essaims du CoLPSO. Cette stratégie est également transposable à d'autres hybridations. En effet, au lieu d'effectuer une hybridation de nature homogène, des hybridations de nature hétérogène sont également envisageables avec d'autres métaheuristiques. Ainsi, un tel processus pourrait être ajouté avant la génération d'une population lorsque les solutions stagnent pour les fourmis provenant de l'algorithme d'optimisation par colonies de fourmis (Alberto Colorni et al., 1991; Dorigo & Gambardella, 1997), ou encore au début d'une nouvelle population d'abeilles provenant de l'algorithme des colonies d'abeilles artificielles (Teodorovic & Dell'Orco, 2005). Ce même processus pourrait être ajouté afin de poursuivre l'exploration de l'espace de recherche, selon le même principe que les algorithmes de recherche locale sont souvent ajoutés aux métaheuristiques pour améliorer l'exploitation des solutions.

Finalement, la modélisation des solutions en anneau proposée au Chapitre 6 est davantage tributaire de la nature du problème et de la modélisation des solutions. C'est cette composante spécifique qui a permis à la métaheuristique d'être plus performante en fonction du problème à résoudre. Cette modélisation est transposable pour le problème classique du voyageur de commerce considérant que l'ordre de visite des villes est très malléable. Elle pourrait difficilement être transposée telle quel pour résoudre des problèmes tels que le RCPSP et le SOP étant donné l'utilisation de contraintes de préséances. La proposition de retirer les bornes du problème, en permettant aux particules du PSO de ne plus être contraintes par les limites de l'espace de recherche, doit être adaptée adéquatement pour qu'elle soit pertinente pour la résolution de problèmes. Également, le nouveau processus d'apprentissage Magnet peut être transposé pour la résolution d'un plus grand nombre de

problèmes. Par exemple, pour le problème de voyageur de commerce, il pourrait être utilisé afin de rapprocher deux villes ayant une courte distance les séparant. Ceci dans le but de diminuer le résultat provenant de l'évaluation par la fonction objectif.

7.3. PERSPECTIVES D'AMÉLIORATION

Le MSA est un problème comportant un très grand nombre d'instances à taille variable. Il s'agit d'un problème présentant un niveau de complexité et il exige des efforts de calculs non négligeables. Même si les instances de MSA utilisées dans cette thèse comportent une dimension inférieure à 1 500, il serait possible d'étendre l'applicabilité à un plus grand nombre d'instances et à des instances de taille encore plus grande. Dans cette thèse, 27 des 38 instances provenant de la catégorie RV11 de BAliBASE 3.0 (Thompson et al., 2005) sont utilisées. Les tests peuvent être étendus afin d'inclure l'entièreté des instances provenant de cet ensemble. D'autres catégories d'instances sont également disponibles avec BAliBASE 3.0, lesquelles proposent des alignements comportant d'autres caractéristiques : des séquences entre 20% et 40% de caractères similaires, l'utilisation d'une séquence orpheline, etc.

Dans un autre ordre d'idées, l'utilisation du calcul haute performance est suggérée afin de contribuer à l'amélioration de la qualité des alignements pour la résolution du MSA (Pais et al., 2014). Considérant la taille des instances et le temps d'exécution nécessaire pour les résoudre, des stratégies de parallélisme permettraient d'améliorer la qualité des solutions en plus de diminuer les temps de résolution. Il s'agit d'une stratégie qu'il est possible d'utiliser afin de résoudre, dans un temps raisonnable, des instances de très grande taille. Une version parallèle utilisant des variantes de PSO a d'ailleurs été proposée par G.-W. Zhang et al. (2015). Cette version combine le GPSO, le LPSO, le CLPSO et le BPSO pour la résolution de fonctions continues. Ces concepts peuvent être transposés pour la résolution du MSA.

Des expérimentations supplémentaires seraient pertinentes afin de contribuer davantage au nouveau processus d'apprentissage Magnet de l'hybride HCoCLPSO-Magnet. Bien qu'une seule version de magnétisme dynamique soit proposée dans cette thèse, d'autres formulations de variations du paramètre de magnétisme Γ pourraient être conçues pour modifier le comportement de la métaheuristique et les opérations de magnétisme lors de la résolution du MSA. Certaines possibilités incluent une adaptation utilisant la dimension du problème, une utilisation des espacements localisés aux colonnes les plus pénalisantes selon l'évaluation par la fonction objectif, ou encore l'ajout d'un processus de recherche locale pour l'espacement résultant du magnétisme.

Certaines stratégies d'amélioration du PSO n'ont pas été abordées dans cette thèse, lesquels proposent une amélioration potentielle de la qualité des solutions. Tout d'abord, il y a l'ajout d'une stratégie d'initialisation des solutions. La conception classique du PSO implique que la composition des particules est initialisée de manière aléatoire selon les variables de l'espace de recherche. Tel qu'il a été démontré par Q. Li et al. (2020), plusieurs métaheuristiques sont sensibles à cette initialisation, ce qui peut impacter grandement l'évolution d'une exécution. Par exemple, concernant la résolution du MSA, les particules du PSO peuvent être initialisées suivant une distribution effectuée en partie en début ou en fin d'une séquence, là où les espacements se regroupent généralement. Une autre option d'initialisation de solutions consiste à utiliser la génération de nombres chaotiques au lieu d'une distribution aléatoire, lequel est un domaine de plus en plus utilisé dans la résolution de problèmes et promet des résultats pertinents selon Senkerik et al. (2015).

Suivant une analyse plus approfondie des expérimentations effectuées au Chapitre 6, il est possible de remarquer que plusieurs particules du PSO deviennent stagnantes après plusieurs générations. La stagnation implique qu'une particule détient une vélocité équivalente à zéro (ou nulle) à un certain moment pendant la résolution. Les tests empiriques

effectués permettent de remarquer que près de 75% des particules deviennent stagnantes après la moitié des évaluations octroyées. Utiliser une stratégie pour réutiliser ces particules stagnantes permettrait de continuer à améliorer la performance du PSO pour la résolution de problèmes.

Les contributions proposées dans cette thèse permettent d'améliorer les stratégies métaheuristiques continues à base de population. La transposition de ces contributions sur davantage de problèmes et de métaheuristiques est une avenue pertinente pour la recherche opérationnelle. Après tout, l'élaboration de nouvelles idées et de nouvelles stratégies n'est limitée que par l'imagination du chercheur.

« What is now proved was once only imagined » - William Blake.

LISTE DE RÉFÉRENCES

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic Algorithms: A Comprehensive Review. Dans A. K. Sangaiah, M. Sheng, & Z. Zhang (Éds.), *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications* (pp. 185-231). Academic Press. https://doi.org/10.1016/B978-0-12-813314-9.00010-4

Akjiratikarl, C., Yenradee, P., & Drake, P. R. (2007). PSO-based algorithm for home care worker scheduling in the UK. *Computers & Industrial Engineering*, *53*(4), 559-583. https://doi.org/10.1016/j.cie.2007.06.002

al-Rifaie, M. M., & Blackwell, T. (2012). *Bare Bones Particle Swarms with Jumps*. International Conference on Swarm Intelligence, Berlin, Heidelberg. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-32650-9_5

Ali, A. B., Luque, G., & Alba, E. (2020). An efficient discrete PSO coupled with a fast local search heuristic for the DNA fragment assembly problem. *Information Sciences*, *512*, 880-908. https://doi.org/10.1016/j.ins.2019.10.026

Anghinolfi, D., Montemanni, R., Paolucci, M., & Maria Gambardella, L. (2011). A hybrid particle swarm optimization approach for the sequential ordering problem. *Computers & Operations Research*, *38*(7), 1076-1085. https://doi.org/10.1016/j.cor.2010.10.014

Anghinolfi, D., & Paolucci, M. (2009). A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, *193*(1), 73-85. https://doi.org/10.1016/j.ejor.2007.10.044

Armougom, F., Moretti, S., Poirot, O., Audic, S., Dumas, P., Schaeli, B., ... Notredame, C. (2006). Expresso: automatic incorporation of structural information in multiple sequence alignments using 3D-Coffee. *Nucleic Acids Research*, *34*(suppl 2), W604-W608. https://doi.org/10.1093/nar/gkl092

Back, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press. https://doi.org/10.1093/oso/9780195099713.001.0001

Bahr, A., Thompson, J. D., Thierry, J. C., & Poch, O. (2001). BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Research*, *29*(1), 323-326. https://doi.org/10.1093/nar/29.1.323

Banks, A., Vincent, J., & Anyakoha, C. (2007). A review of particle swarm optimization. Part I: background and development. *Natural Computing*, *6*(4), 467-484. https://doi.org/10.1007/s11047-007-9049-

Banks, A., Vincent, J., & Anyakoha, C. (2008). A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. *Natural Computing*, *7*(1), 109-124. https://doi.org/10.1007/s11047-007-9050-z

Barzilay, R., & Lee, L. (2003). *Learning To Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment*. Proceedings of HLT-NAACL. Association for Computational Linguistics. https://arxiv.org/abs/cs/0304006

Bawono, P., Dijkstra, M., Pirovano, W., Feenstra, A., Abeln, S., & Heringa, J. (2017). Multiple Sequence Alignment. Dans J. M. Keith (Éd.), *Bioinformatics: Volume 1: Data, Sequence Analysis, and Evolution* (pp. 167-189). Springer New York. https://doi.org/10.1007/978-1-4939-6622-6_8

Blum, C., Puchinger, J., Raidl, G. R., & Roli, A. (2011). Hybrid metaheuristics in combinatorial optimization: A survey. *Applied Soft Computing*, *11*(6), 4135-4151. https://doi.org/10.1016/j.asoc.2011.02.032

Boese, K. D., Kahng, A. B., & Muddu, S. (1994). A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters*, *16*(2), 101-113. https://doi.org/10.1016/0167-6377(94)90065-5

Botta, M., & Negro, G. (2010). Multiple Sequence Alignment with Genetic Algorithms. Dans F. Masulli, L. E. Peterson, & R. Tagliaferri (Éds.), *International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics* (pp. 206-214). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-14571-1_15

Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, *237*, 82-117. https://doi.org/10.1016/j.ins.2013.02.041

Chellapilla, K., & Fogel, G. B. (1999, 6-9 July 1999). *Multiple sequence alignment using evolutionary programming*. Congress on Evolutionary Computation, Washington, DC, USA. IEEE. https://doi.org/10.1109/CEC.1999.781958

Chen, R.-M., Wu, C.-L., Wang, C.-M., & Lo, S.-T. (2010). Using novel particle swarm optimization scheme to solve resource-constrained scheduling problem in PSPLIB. *Expert systems with applications*, *37*(3), 1899-1910. https://doi.org/10.1016/j.eswa.2009.07.024

Chen, T., Zhang, B., Hao, X., & Dai, Y. (2006). *Task Scheduling in Grid Based on Particle Swarm Optimization*. International Symposium on Parallel and Distributed Computing. IEEE. https://doi.org/10.1109/ISPDC.2006.46

Clerc, M. (1999). *The swarm and the queen: towards a deterministic and adaptive particle swarm optimization*. Congress on Evolutionary Computation, Washington, DC, USA. IEEE. https://doi.org/10.1109/CEC.1999.785513

Clerc, M. (2004). Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem. Dans G. C. Onwubolu, & B. V. Babu (Éds.), *New Optimization Techniques in Engineering* (Vol. 141, pp. 219-239). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-39930-8_8

Clerc, M. (2006). *Particle swarm optimization* (Vol. 93). John Wiley & Sons.

Collette, Y., & Siarry, P. (2004). *Multiobjective Optimization: Principles and Case Studies*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-08883-8

Colorni, A., Dorigo, M., Maffioli, F., Maniezzo, V., Righini, G., & Trubian, M. (1996). Heuristics from nature for hard combinatorial optimization problems. *International Transactions in Operational Research*, *3*(1), 1-21. https://doi.org/10.1016/0969-6016(96)00004-4

Colorni, A., Dorigo, M., & Maniezzo, V. (1991). *Distributed optimization by ant colonies*. Proceedings of the First European Conference on Artificial Life, Paris, France.

Cuevas, E., Zaldivar, D., Pérez-Cisneros, M., & Ramírez-Ortegón, M. (2011). Circle detection using discrete differential evolution optimization. *Pattern Analysis and Applications*, *14*(1), 93-107. https://doi.org/10.1007/s10044-010-0183-9

Dayhoff, M., Schwartz, R., & Orcutt, B. (1978). A model of evolutionary change in proteins. Dans M. Dayhoff (Éd.), *Atlas of Protein Sequence and Structure* (Vol. 5, pp. 345-352). National Biomedical Research Foundation.

Dayhoff, M. O. (1972). *Atlas of protein sequence and structure*. National Biomedical Research Foundation.

Del Valle, Y., Venayagamoorthy, G. K., Mohagheghi, S., Hernandez, J.-C., & Harley, R. G. (2008). Particle swarm optimization: basic concepts, variants and applications in power systems. *IEEE Transactions on Evolutionary Computation*, *12*(2), 171-195. https://doi.org/10.1109/TEVC.2007.896686

Deschenes, H., & Gagne, C. (2020). Solving high dimensional multimodal continuous optimisation problems using hybridisation between particle swarm optimisation variants. *International Journal of Metaheuristics*, *7*(3), 239-264. https://doi.org/10.1504/ijmheur.2020.107390

Dixon, L. C. W., & Szego, G. P. (1978). The Global Optimization Problem: An Introduction. Dans L. C. W. Dixon, & G. P. Szego (Éds.), *Towards Global Optimization* (Vol. 2, pp. 1-15).

Do, C. B., Mahabhashyam, M. S., Brudno, M., & Batzoglou, S. (2005). ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Research*, *15*(2), 330-340. https://doi.org/10.1101/gr.2821705

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53-66. https://doi.org/10.1109/4235.585892

Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and Unsupervised Discretization of Continuous Features. Dans 10.1016/B978-1-55860-377-6.50032-3 (Éd.), *Machine learning: proceedings 1995* (pp. 194-202). Morgan Kaufmann. https://doi.org/10.1016/B978-1-55860-377-6.50032-3

Du, S.-Y., & Liu, Z.-G. (2020). Hybridizing Particle Swarm Optimization with JADE for continuous optimization. *Multimedia Tools and Applications*, *79*(7), 4619-4636. https://doi.org/10.1007/s11042-019-08142-7

Eberhart, R., & Kennedy, J. (1995, 4-6 Oct 1995). *A new optimizer using particle swarm theory*. IEEE International Symposium on Micro Machine and Human Science, Nagoya, Japan. IEEE. https://doi.org/10.1109/MHS.1995.494215

Eberhart, R. C., & Shi, Y. (2000). *Comparing inertia weights and constriction factors in particle swarm optimization*. IEEE Congress on Evolutionary Computation. IEEE. https://doi.org/10.1109/CEC.2000.870279

Eberhart, R. C., & Shi, Y. (2001). *Particle swarm optimization: developments, applications and resources*. IEEE Congress on Evolutionary Computation. IEEE. https://doi.org/10.1109/CEC.2001.934374

Edgar, R. C. (2004a). MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, *5*(1), 113. https://doi.org/10.1186/1471-2105-5-113

Edgar, R. C. (2004b). MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, *32*(5), 1792-1797. https://doi.org/10.1093/nar/gkh340

Escudero, L. F. (1988). An inexact algorithm for the sequential ordering problem. *European Journal of Operational Research*, *37*(2), 236-249. https://doi.org/10.1016/0377-2217(88)90333-5

Farmer, J. D., Packard, N. H., & Perelson, A. S. (1986). The immune system, adaptation, and machine learning. *Physica D: Nonlinear Phenomena*, *22*(1-3), 187-204. https://doi.org/10.1016/0167-2789(86)90240-X

Fister Jr, I., Yang, X.-S., Fister, I., Brest, J., & Fister, D. (2013). A Brief Review of Nature-Inspired Algorithms for Optimization. *Elektrotehniski Vestnik*, *80*, 116-122. https://arxiv.org/abs/1307.4186

Gao, L., & Hailu, A. (2010). Comprehensive Learning Particle Swarm Optimizer for Constrained Mixed-Variable Optimization Problems. *International Journal of Computational Intelligence Systems*, *3*(6), 832-842. https://doi.org/10.1080/18756891.2010.9727745

García-Nieto, J., & Alba, E. (2012). *Why six informants is optimal in pso*. Annual conference on Genetic and evolutionary computation conference. ACM. https://doi.org/10.1145/2330163.2330168

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of Np-Completeness* (Vol. 174). W. H. Freeman and Company.

Gass1, S. I., & Fu, M. C. (2013). *Encyclopedia of Operations Research and Management Science*. Springer Boston MA. https://doi.org/10.1007/978-1-4419-1153-7

Ge, H.-W., & Liang, Y.-C. (2005). *A Hidden Markov Model and Immune Particle Swarm Optimization-Based Algorithm for Multiple Sequence Alignment*. AI2005: Advances in Artificial Intelligence, Berlin, Heidelberg. Springer Berlin Heidelberg. https://doi.org/10.1007/11589990_78

Glover, F. (1989). Tabu search-part I. *ORSA Journal on computing*, *1*(3), 190-206. https://doi.org/10.1287/ijoc.1.3.190

Glover, F. (1990). Tabu search—part II. *ORSA Journal on computing*, *2*(1), 4-32. https://doi.org/10.1287/ijoc.2.1.4

Goldberg, D., & Holland, J. (1988). Genetic Algorithms and Machine Learning. *Machine Learning*, *3*(2-3), 95-99. https://doi.org/10.1023/A:1022602019183

Henikoff, S., & Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, *89*(22), 10915-10919. https://doi.org/10.1073/pnas.89.22.10915

Ho, Y.-C., & Pepyne, D. L. (2002). Simple Explanation of the No-Free-Lunch Theorem and Its Implications. *Journal of Optimization Theory and Applications*, *115*(3), 549-570. https://doi.org/10.1023/A:1021251113462

Hung, C.-L., Lin, Y.-S., Lin, C.-Y., Chung, Y.-C., & Chung, Y.-F. (2015). CUDA ClustalW: An efficient parallel algorithm for progressive multiple sequence alignment on Multi-GPUs. *Computational Biology and Chemistry*, *58*, 62-68. https://doi.org/10.1016/j.compbiolchem.2015.05.004

Hussain, K., Salleh, M. N. M., Cheng, S., & Shi, Y. (2019). Metaheuristic research: a comprehensive survey. *Artificial Intelligence Review*, *52*(4), 2191-2233. https://doi.org/10.1007/s10462-017-9605-z

Issa, M., & Hassanien, A. E. (2020). Multiple Sequence Alignment Optimization Using Meta-Heuristic Techniques. Dans *Data Analytics in Medicine: Concepts, Methodologies, Tools, and Applications* (pp. 565-579). IGI Global. https://doi.org/10.4018/978-1-7998-1204-3.ch031

Jäger, G. (2015). Support for linguistic macrofamilies from weighted sequence alignment. *Proceedings of the National Academy of Sciences*, *112*(41), 12752-12757. https://doi.org/10.1073/pnas.1500331112

Jarboui, B., Damak, N., Siarry, P., & Rebai, A. (2008). A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. *Applied Mathematics and Computation*, *195*(1), 299-308. https://doi.org/10.1016/j.amc.2007.04.096

Jarboui, B., Ibrahim, S., Siarry, P., & Rebai, A. (2008). A combinatorial particle swarm optimisation for solving permutation flowshop problems. *Computers & Industrial Engineering*, *54*(3), 526-538. https://doi.org/10.1016/j.cie.2007.09.006

Jia, Q., & Seo, Y. (2013). An improved particle swarm optimization for the resource-constrained project scheduling problem. *The International Journal of Advanced Manufacturing Technology*, *67*(9-12), 2627-2638. https://doi.org/10.1007/s00170-012-4679-x

Jiang, B., & Wang, N. (2014). Cooperative bare-bone particle swarm optimization for data clustering. *Soft Computing*, *18*(6), 1079-1091. https://doi.org/10.1007/s00500-013-1128-1

Jiang, F., Xia, H., Anh Tran, Q., Minh Ha, Q., Quang Tran, N., & Hu, J. (2017). A new binary hybrid particle swarm optimization with wavelet mutation. *Knowledge-Based Systems*, *130*, 90-101. https://doi.org/10.1016/j.knosys.2017.03.032

Jin, Y.-X., Cheng, H.-Z., Yan, J.-y., & Zhang, L. (2007). New discrete method for particle swarm optimization and its application in transmission network expansion planning. *Electric Power Systems Research*, *77*(3–4), 227-233. https://doi.org/10.1016/j.epsr.2006.02.016

Jordehi, A. R., & Jasni, J. (2015). Particle swarm optimisation for discrete optimisation problems: a review. *Artificial Intelligence Review*, *43*(2), 243-258. https://doi.org/10.1007/s10462-012-9373-8

Jourdan, L., Basseur, M., & Talbi, E.-G. (2009). Hybridizing exact methods and metaheuristics: A taxonomy. *European Journal of Operational Research*, *199*(3), 620-629. https://doi.org/10.1016/j.ejor.2007.07.035

Juang, W. S., & Su, S. F. (2008). Multiple sequence alignment using modified dynamic programming and particle swarm optimization. *Journal of the Chinese Institute of Engineers*, *31*(4), 659-673. https://doi.org/10.1080/02533839.2008.9671419

Kamal, A., Mahroos, M., Sayed, A., & Nassar, A. (2012). Parallel Particle Swarm Optimization for Global Multiple Sequence Alignment. *Information Technology Journal*, *11*(8), 998-1006. https://doi.org/10.3923/itj.2012.998.1006

Kartous, W., Layeb, A., & Chikhi, S. (2014). A new quantum cuckoo search algorithm for multiple sequence alignment. *Journal of Intelligent Systems*, *23*(3), 261-275. https://doi.org/10.1515/jisys-2013-0052

Katoh, K., Misawa, K., Kuma, K. i., & Miyata, T. (2002). MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Research*, *30*(14), 3059-3066. https://doi.org/10.1093/nar/gkf436

Katoh, K., & Standley, D. M. (2016). A simple method to control over-alignment in the MAFFT multiple sequence alignment program. *Bioinformatics*, *32*(13), 1933-1942. https://doi.org/10.1093/bioinformatics/btw108

Kaya, M., Sarhan, A., & Alhajj, R. (2014). Multiple sequence alignment with affine gap by using multi-objective genetic algorithm. *Computer Methods and Programs in Biomedicine*, *114*(1), 38-49. https://doi.org/10.1016/j.cmpb.2014.01.013

Kennedy, J. (2003). *Bare bones particle swarms*. IEEE Swarm Intelligence Symposium. IEEE. https://doi.org/10.1109/SIS.2003.1202251

Kennedy, J., & Eberhart, R. C. (1997, 12-15 Oct 1997). *A discrete binary version of the particle swarm algorithm*. IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, Orlando, FL, USA. IEEE. https://doi.org/10.1109/ICSMC.1997.637339

Kennedy, J., & Mendes, R. (2002, 12-17 May 2002). *Population structure and particle swarm performance*. IEEE Congress on Evolutionary Computation, Honolulu, HI, USA. IEEE. https://doi.org/10.1109/CEC.2002.1004493

Kim, J., Pramanik, S., & Chung, M. J. (1994). Multiple sequence alignment using simulated annealing. *Bioinformatics*, *10*(4), 419-426. https://doi.org/10.1093/bioinformatics/10.4.419

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, *220*(4598), 671-680. https://doi.org/10.1126/science.220.4598.671

Kolisch, R., & Sprecher, A. (1997). PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program. *European*

Journal of Operational Research, *96*(1), 205-216. https://doi.org/10.1016/S0377-2217(96)00170-1

Krause, J., Cordeiro, J., Parpinelli, R. S., & Lopes, H. S. (2013). A survey of swarm algorithms applied to discrete optimization problems. *Swarm Intelligence and Bio-Inspired Computation*, *4*(9), 169-191. https://doi.org/10.1016/B978-0-12-405163-8.00007-7

Krohling, R., & Mendel, E. (2009, 18-21 May 2019). *Bare bones particle swarm optimization with Gaussian or Cauchy jumps*. IEEE Congress on Evolutionary Computation. IEEE. https://doi.org/10.1109/CEC.2009.4983361

Kumar, M. (2015). An enhanced algorithm for multiple sequence alignment of protein sequences using genetic algorithm. *EXCLI Journal*, *14*, 1232-1255. https://doi.org/10.17179/excli2015-302

Lalwani, S., Kumar, R., & Gupta, N. (2013). A review on particle swarm optimization variants and their applications to multiple sequence alignment. *Journal of Applied Mathematics and Bioinformatics*, *3*(2), 87-124.

Lalwani, S., Kumar, R., & Gupta, N. (2015). A novel two-level particle swarm optimization approach for efficient multiple sequence alignment. *Memetic Computing*, *7*(2), 119-133. https://doi.org/10.1007/s12293-015-0157-y

Larkin, M. A., Blackshields, G., Brown, N. P., Chenna, R., McGettigan, P. A., McWilliam, H., ... Higgins, D. G. (2007). Clustal W and Clustal X version 2.0. *Bioinformatics*, *23*(21), 2947-2948. https://doi.org/10.1093/bioinformatics/btm404

Laskari, E. C., Parsopoulos, K. E., & Vrahatis, M. N. (2002, 12-17 May 2002). *Particle swarm optimization for integer programming*. IEEE Congress on Evolutionary Computation, Honolulu, HI, USA. IEEE. https://doi.org/10.1109/CEC.2002.1004478

Lee, Z.-J., Su, S.-F., Chuang, C.-C., & Liu, K.-H. (2008). Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Applied Soft Computing*, *8*(1), 55-78. https://doi.org/10.1016/j.asoc.2006.10.012

Lei, X.-j., Sun, J.-j., & Ma, Q.-z. (2009). Multiple Sequence Alignment Based on Chaotic PSO. Dans Z. Cai, Z. Li, Z. Kang, & Y. Liu (Éds.), *Computational Intelligence and Intelligent Systems: 4th International Symposium, ISICA 2009, Huangshi, China, October 23-25, 2009. Proceedings* (pp. 351-360). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-04962-0_40

Lemamou, E., Gagné, C., & Gravel, M. (2010). *Optimisation par essaims particulaires pour le problème RCPSP*. Conférence Internationale de MOdélisation et SIMulation, Hammamet, Tunisie.

Li, Q., Liu, S.-Y., & Yang, X.-S. (2020). Influence of initialization on the performance of metaheuristic optimizers. *Applied Soft Computing*, *91*, 106193. https://doi.org/10.1016/j.asoc.2020.106193

Li, X., & Yao, X. (2012). Cooperatively coevolving particle swarms for large scale optimization. *IEEE Transactions on Evolutionary Computation*, *16*(2), 210-224. https://doi.org/10.1109/TEVC.2011.2112662

Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation*, *10*(3), 281-295. https://doi.org/10.1109/TEVC.2005.857610

Liao, C.-J., Tseng, C.-T., & Luarn, P. (2007). A discrete version of particle swarm optimization for flowshop scheduling problems. *Computers & Operations Research*, *34*(10), 3099-3111. https://doi.org/10.1016/j.cor.2005.11.017

Liu, G., Chen, X., Zhou, R., Xu, S., Chen, Y.-C., & Chen, G. (2021). Social learning discrete Particle Swarm Optimization based two-stage X-routing for IC design under Intelligent Edge Computing architecture. *Applied Soft Computing*, *104*, 107215. https://doi.org/10.1016/j.asoc.2021.107215

Liu, G., Zhu, W., Xu, S., Zhuang, Z., Chen, Y.-C., & Chen, G. (2020). Efficient VLSI routing algorithm employing novel discrete PSO and multi-stage transformation. *Journal of Ambient Intelligence and Humanized Computing*, 1-16. https://doi.org/10.1007/s12652-020-02659-8

Long, H. X., Xu, W. B., Sun, J., & Ji, W. J. (2009, 14-16 Aug. 2009). *Multiple Sequence Alignment Based on a Binary Particle Swarm Optimization Algorithm*. IEEE International Conference on Natural Computation, Tianjian, China. IEEE. https://doi.org/10.1109/ICNC.2009.238

Mahadevan, K., & Kannan, P. (2010). Comprehensive learning particle swarm optimization for reactive power dispatch. *Applied Soft Computing*, *10*(2), 641-652. https://doi.org/10.1016/j.asoc.2009.08.038

Mahdavi, S., Shiri, M. E., & Rahnamayan, S. (2015). Metaheuristics in large-scale global continues optimization: A survey. *Information Sciences*, *295*, 407-428. https://doi.org/10.1016/j.ins.2014.10.042

Maitra, M., & Chatterjee, A. (2008). A hybrid cooperative–comprehensive learning based PSO algorithm for image segmentation using multilevel thresholding. *Expert Systems with Applications*, *34*(2), 1341-1350. https://doi.org/10.1016/j.eswa.2007.01.002

Mavrovouniotis, M., Li, C., & Yang, S. (2017). A survey of swarm intelligence for dynamic optimization: Algorithms and applications. *Swarm and Evolutionary Computation*, *33*, 1-17. https://doi.org/10.1016/j.swevo.2016.12.005

Miao, Z., Yong, P., Mei, Y., Quanjun, Y., & Xu, X. (2021). A discrete PSO-based static load balancing algorithm for distributed simulations in a cloud environment. *Future Generation Computer Systems*, *115*, 497-516. https://doi.org/10.1016/j.future.2020.09.016

Mohamed, I., & Aboul Ella, H. (2017). Multiple Sequence Alignment Optimization Using Meta-Heuristic Techniques. Dans H. Aboul Ella, & G. Tarek (Éds.), *Handbook of Research on Machine Learning Innovations and Trends* (pp. 409-423). IGI Global. https://doi.org/10.4018/978-1-5225-2229-4.ch018

Molga, M., & Smutnicki, C. (2005). Test functions for optimization needs. *Test functions for optimization needs*, *101*.

Mondal, S. K., Chatterjee, A., & Tudu, B. (2018, 2018//). *A Hybrid Particle Swarm Optimization and Artificial Bee Colony Algorithm for Image Contrast Enhancement*. Proceedings of the International Conference on Computing and Communication Systems, Singapore. Springer Singapore. https://doi.org/10.1007/978-981-10-6890-4_26

Moustafa, N., Elhosseini, M., Taha, T. H., & Salem, M. (2017). Fragmented protein sequence alignment using two-layer particle swarm optimization (FTLPSO). *Journal of King Saud University - Science*, *29*(2), 191-205. https://doi.org/10.1016/j.jksus.2016.04.007

Naznin, F., Sarker, R., & Essam, D. (2011). Vertical decomposition with genetic algorithm for multiple sequence alignment. *BMC Bioinformatics*, *12*(1), 353. https://doi.org/10.1186/1471-2105-12-353

Notredame, C. (2004). Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics*, *3*(1), 131-144. https://doi.org/10.1517/14622416.3.1.131

Notredame, C., Higgins, D. G., & Heringa, J. (2000). T-coffee: a novel method for fast and accurate multiple sequence alignment1. *Journal of Molecular Biology*, *302*(1), 205-217. https://doi.org/10.1006/jmbi.2000.4042

Onwubolu, G. C., & Davendra, D. (2009). *Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization* (Vol. 175). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-92151-6

Osman, I. H., & Kelly, J. P. (1996). Meta-Heuristics: An Overview. Dans I. H. Osman, & J. P. Kelly (Éds.), *Meta-Heuristics: Theory and Applications* (pp. 1-21). Springer US. https://doi.org/10.1007/978-1-4613-1361-8_1

Öztürk, C., & Aslan, S. (2016). A new artificial bee colony algorithm to solve the multiple sequence alignment problem. *International Journal of Data Mining and Bioinformatics*, *14*(4), 332-353. https://doi.org/10.1504/ijdmb.2016.075823

Pais, F. S.-M., Ruy, P. d., Oliveira, G., & Coimbra, R. (2014). Assessing the efficiency of multiple sequence alignment programs. *Algorithms for Molecular Biology*, *9*. https://doi.org/10.1186/1748-7188-9-4

Pan, Q.-K., Tasgetiren, M. F., & Liang, Y.-C. (2008). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Comput. Oper. Res.*, *35*(9), 2807-2839. https://doi.org/10.1016/j.cor.2006.12.030

Pang, W., Wang, K.-p., Zhou, C.-g., & Dong, L.-j. (2004, 2004/10/16). *Fuzzy discrete particle swarm optimization for solving traveling salesman problem*. The Fourth International Conference on Computer and Information Technology, Wuhan, China. IEEE. https://doi.org/10.1109/CIT.2004.1357292

Pearson, W. R. (1996). Effective protein sequence comparison. Dans *Methods in enzymology* (Vol. 266, pp. 227-258). Academic Press. https://doi.org/10.1016/s0076-6879(96)66017-0

Pervez, M. T., Babar, M. E., Nadeem, A., Aslam, M., Awan, A. R., Aslam, N., ... Shoaib, M. (2014). Evaluating the Accuracy and Efficiency of Multiple Sequence Alignment Methods. *Evolutionary Bioinformatics Online*, *10*, 205-217. https://doi.org/10.4137/EBO.S19199

Pevsner, J. (2015). *Bioinformatics and functional genomics*. John Wiley & Sons.

Pluhacek, M., Kadavy, T., Senkerik, R., Viktorin, A., & Zelinka, I. (2016, 6-9 Dec. 2016). *Comparing selected PSO modifications on CEC 15 benchmark set*. IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece. IEEE. https://doi.org/10.1109/SSCI.2016.7850279

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. *Swarm intelligence*, *1*(1), 33-57. https://doi.org/10.1007/s11721-007-0002-0

Ponnambalam, S., Jawahar, N., & Chandrasekaran, S. (2009). *Discrete particle swarm optimization algorithm for flowshop scheduling*. INTECH Open Access Publisher.

Pritsker, A. A. B., Waiters, L. J., & Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science*, *16*(1), 93-108. https://doi.org/10.1287/mnsc.16.1.93

Prokić, J., Wieling, M., & Nerbonne, J. (2009, 2009/03/30). *Multiple sequence alignments in linguistics*. Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education, Athens, Greece. Association for Computational Linguistics.

Puchinger, J., & Raidl, G. R. (2005). *Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification*. Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Berlin, Heidelberg. Springer Berlin Heidelberg. https://doi.org/10.1007/11499305_5

Puranik, P., Bajaj, P., Abraham, A., Palsodkar, P., & Deshmukh, A. (2009). *Human Perception-Based Color Image Segmentation Using Comprehensive Learning Particle Swarm Optimization*. IEEE International Conference on Emerging Trends in Engineering & Technology. IEEE. https://doi.org/10.1109/ICETET.2009.116

Qu, B. Y., Liang, J. J., Wang, Z. Y., Chen, Q., & Suganthan, P. N. (2016). Novel benchmark functions for continuous multimodal optimization with comparative results. *Swarm and Evolutionary Computation*, *26*, 23-34. https://doi.org/10.1016/j.swevo.2015.07.003

Raidl, G. R. (2006). A Unified View on Hybrid Metaheuristics. Dans F. Almeida, M. J. Blesa Aguilera, C. Blum, J. M. Moreno Vega, M. Pérez Pérez, A. Roli, & M. Sampels (Éds.), *Hybrid Metaheuristics* (Vol. 4030, pp. 1-12). Springer Berlin Heidelberg. https://doi.org/10.1007/11890584_1

Raidl, G. R. (2015). Decomposition based hybrid metaheuristics. *European Journal of Operational Research*, *244*(1), 66-76. https://doi.org/10.1016/j.ejor.2014.12.005

Rasmussen, T. K., & Krink, T. (2003). Improved Hidden Markov Model training for multiple sequence alignment by a particle swarm optimization—evolutionary algorithm hybrid. *Biosystems*, *72*(1–2), 5-17. https://doi.org/10.1016/S0303-2647(03)00131-X

Ratnaweera, A., Halgamuge, S., & Watson, H. C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, *8*(3), 240-255. https://doi.org/10.1109/TEVC.2004.826071

Ravindran, A. R. (2016). *Operations research and management science handbook*. Crc Press.

Richer, T. J., & Blackwell, T. M. (2006). *The Lévy particle swarm*. IEEE International Conference on Evolutionary Computation. IEEE. https://doi.org/10.1109/CEC.2006.1688394

Rubio-Largo, Á., Vega-Rodríguez, M. A., & González-Álvarez, D. L. (2016). Hybrid multiobjective artificial bee colony for multiple sequence alignment. *Applied Soft Computing*, *41*, 157-168. https://doi.org/10.1016/j.asoc.2015.12.034

Salman, A., Ahmad, I., & Al-Madani, S. (2002). Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, *26*(8), 363-371. https://doi.org/10.1016/S0141-9331(02)00053-4

Schutte, J. F., & Groenwold, A. A. (2005). A Study of Global Optimization Using Particle Swarms. *Journal of Global Optimization*, *31*(1), 93-108. https://doi.org/10.1007/s10898-003-6454-x

Sengupta, S., Basak, S., & Peters, R. A. (2018). Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives. *Machine Learning and Knowledge Extraction*, *1*, 157-191. https://doi.org/10.3390/make1010010

Senkerik, R., Pluhacek, M., Zelinka, I., Davendra, D., & Oplatkova, Z. K. (2015). A Brief Survey on the Chaotic Systems as the Pseudo Random Number Generators. Dans A. Sanayei, O. E. Rössler, & I. Zelinka (Éds.), *Interdisciplinary Symposium on Complex Systems* (pp. 205-214). Springer Cham. https://doi.org/10.1007/978-3-319-10759-2_22

Sha, D., & Hsu, C.-Y. (2006). A hybrid particle swarm optimization for job shop scheduling problem. *Computers & Industrial Engineering*, *51*(4), 791-808. https://doi.org/10.1016/j.cie.2006.09.002

Shayeghi, H., Mahdavi, M., & Bagheri, A. (2010a). Discrete PSO algorithm based optimization of transmission lines loading in TNEP problem. *Energy Conversion and Management*, *51*(1), 112-121. https://doi.org/10.1016/j.enconman.2009.08.030

Shayeghi, H., Mahdavi, M., & Bagheri, A. (2010b). An improved DPSO with mutation based on similarity algorithm for optimization of transmission lines loading. *Energy Conversion and Management*, *51*(12), 2715-2723. https://doi.org/10.1016/j.enconman.2010.06.007

Shelokar, P. S., Siarry, P., Jayaraman, V. K., & Kulkarni, B. D. (2007). Particle swarm and ant colony algorithms hybridized for improved continuous optimization. *Applied Mathematics and Computation*, *188*(1), 129-142. https://doi.org/10.1016/j.amc.2006.09.098

Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C., & Wang, Q. (2007). Particle swarm optimization-based algorithms for TSP and generalized TSP. *Information Processing Letters*, *103*(5), 169-176. https://doi.org/10.1016/j.ipl.2007.03.010

Shi, Y., & Eberhart, R. C. (1998, 4-9 May 1998). *A Modified Particle Swarm Optimizer*. IEEE International Conference on Evolutionary Computation. IEEE. https://doi.org/10.1109/ICEC.1998.699146

Soleimani, H., & Kannan, G. (2015). A hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks.

Applied Mathematical Modelling, *39*(14), 3990-4012. https://doi.org/10.1016/j.apm.2014.12.016

Storn, R., & Price, K. (1997). Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, *11*(4), 341-359. https://doi.org/10.1023/A:1008202821328

Strnad, D., & Kohek, Š. (2017). Novel discrete differential evolution methods for virtual tree pruning optimization. *Soft Computing*, *21*(4), 981-993. https://doi.org/10.1007/s00500-015-1827-x

Sun, J., Palade, V., Wu, X., & Fang, W. (2014). Multiple Sequence Alignment with Hidden Markov Models Learned by Random Drift Particle Swarm Optimization. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *11*(1), 243-257. https://doi.org/10.1109/TCBB.2013.148

Sun, J., Wu, X., Fang, W., Ding, Y., Long, H., & Xu, W. (2012). Multiple sequence alignment using the Hidden Markov Model trained by an improved quantum-behaved particle swarm optimization. *Information Sciences*, *182*(1), 93-114. https://doi.org/10.1016/j.ins.2010.11.014

Sun, Y., Wang, Z., Qi, G., & Van Wyk, B. J. (2011). Chaotic particle swarm optimization with neural network structure and its application. *Engineering Optimization*, *43*(1), 19-37. https://doi.org/10.1080/03052151003702604

Talbi, E.-G. (2002). A Taxonomy of Hybrid Metaheuristics. *Journal of Heuristics*, *8*(5), 541-564. https://doi.org/10.1023/A:1016540724870

Talbi, E.-G. (2009). *Metaheuristics: from design to implementation* (Vol. 74). John Wiley & Sons.

Tamayo-Vera, D., Chen, S., Bolufé-Röhler, A., Montgomery, J., & Hendtlass, T. (2018). *Improved Exploration and Exploitation in Particle Swarm Optimization*. International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Cham. Springer, Cham. https://doi.org/10.1007/978-3-319-92058-0_41

Tasgetiren, M. F., Suganthan, P. N., & Pan, Q.-Q. (2007). *A discrete particle swarm optimization algorithm for the generalized traveling salesman problem*. Proceedings of the 9th annual conference on Genetic and evolutionary computation, London, England. https://doi.org/10.1145/1276958.1276980

Taylor, W. R. (1987). Multiple sequence alignment by a pairwise algorithm. *Bioinformatics*, *3*(2), 81-87. https://doi.org/10.1093/bioinformatics/3.2.81

Teodorovic, D., & Dell'Orco, M. (2005). Bee colony optimization–a cooperative learning approach to complex transportation problems. *Advanced OR and AI Methods in Transportation*, 51-60.

Thompson, J. D., Koehl, P., Ripp, R., & Poch, O. (2005). BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins: Structure, Function, and Bioinformatics*, *61*(1), 127-136. https://doi.org/10.1002/prot.20527

Tian, Y., Liu, D., Yuan, D., & Wang, K. (2013). A discrete PSO for two-stage assembly scheduling problem. *The International Journal of Advanced Manufacturing Technology*, *66*(1), 481-499. https://doi.org/10.1007/s00170-012-4343-5

van den Bergh, F., & Engelbrecht, A. P. (2004). A Cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 225-239. https://doi.org/10.1109/TEVC.2004.826069

Van den Bergh, F., & Engelbrecht, A. P. (2006). A study of particle swarm optimization particle trajectories. *Information Sciences*, *176*(8), 937-971. https://doi.org/10.1016/j.ins.2005.02.003

Wang, G.-G., Gandomi, A. H., Alavi, A. H., & Deb, S. (2016). A hybrid method based on krill herd and quantum-behaved particle swarm optimization. *Neural Computing and Applications*, *27*(4), 989-1006. https://doi.org/10.1007/s00521-015-1914-z

Wang, L., & Jiang, T. (1994). On the Complexity of Multiple Sequence Alignment. *Journal of Computational Biology*, *1*(4), 337-348. https://doi.org/10.1089/cmb.1994.1.337

Wang, L., Pan, Q.-K., Suganthan, P. N., Wang, W.-H., & Wang, Y.-M. (2010). A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems. *Computers & Operations Research*, *37*(3), 509-520. https://doi.org/10.1016/j.cor.2008.12.004

Wang, W., Wang, H., & Rahnamayan, S. (2011). Improving comprehensive learning particle swarm optimiser using generalised opposition-based learning. *International Journal of Modelling, Identification and Control*, *14*(4), 310-316. https://doi.org/10.1504/ijmic.2011.043155

Wang, X., Mu, A., & Zhu, S. (2013). ISPO: A New Way to Solve Traveling Salesman Problem. *Intelligent Control & Automation*, *4*(2), 122-125. https://doi.org/10.4236/ica.2013.42017

Weise, T., Zapf, M., Chiong, R., & Nebro, A. J. (2009). Why Is Optimization Difficult? Dans R. Chiong (Éd.), *Nature-Inspired Algorithms for Optimisation* (Vol. 193, pp. 1-50). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-00267-0_1

Whitacre, J. M. (2011). Survival of the flexible: explaining the recent popularity of nature-inspired optimization within a rapidly evolving world. *Computing*, *93*(2), 135-146. https://doi.org/10.1007/s00607-011-0156-x

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82. https://doi.org/10.1109/4235.585893

Wu, J., Long, J., & Liu, M. (2015). Evolving RBF neural networks for rainfall prediction using hybrid particle swarm optimization and genetic algorithm. *Neurocomputing*, *148*, 136-142. https://doi.org/10.1016/j.neucom.2012.10.043

Wu, Z., Ni, Z., Zhang, C., & Gu, L. (2008). *Opposition based comprehensive learning particle swarm optimization*. Intelligent System and Knowledge Engineering, 2008. ISKE 2008. 3rd International Conference on. IEEE.

Xing, B., & Gao, W.-J. (2014). *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms* (1st ed. 2014., Vol. 62). Springer Cham. https://doi.org/10.1007/978-3-319-03404-1

Xiong, T., Bao, Y., Hu, Z., & Chiong, R. (2015). Forecasting interval time series using a fully complex-valued RBF neural network with DPSO and PSO algorithms. *Information Sciences*, *305*, 77-92. https://doi.org/10.1016/j.ins.2015.01.029

Xu, F., & Chen, Y. (2009). *A Method for Multiple Sequence Alignment Based on Particle Swarm Optimization*. International Conference on Intelligent Computing, Berlin, Heidelberg. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-04020-7_104

Yang, X.-S. (2010). A New Metaheuristic Bat-Inspired Algorithm. Dans J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, & N. Krasnogor (Éds.), *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)* (pp. 65-74). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12538-6_6

Yu, C., Kelley, L. C., & Tan, Y. (2015, 25-28 May 2015). *Dynamic search fireworks algorithm with covariance mutation for solving the CEC 2015 learning based competition problems*. IEEE Congress on Evolutionary Computation, Sendai, Japan. IEEE. https://doi.org/10.1109/CEC.2015.7257013

Yusoff, M., Ariffin, J., & Mohamed, A. (2015). DPSO based on a min-max approach and clamping strategy for the evacuation vehicle assignment problem. *Neurocomputing*, *148*, 30-38. https://doi.org/10.1016/j.neucom.2012.12.083

Zhang, G.-W., Zhan, Z.-H., Du, K.-J., Lin, Y., Chen, W.-N., Li, J.-J., & Zhang, J. (2015). Parallel Particle Swarm Optimization Using Message Passing Interface. Dans H. Handa, H. Ishibuchi, Y.-S. Ong, & K. C. Tan (Éds.), *Proceedings of the 18th Asia*

Pacific Symposium on Intelligent and Evolutionary Systems (Vol. 1, pp. 55-64). Springer International Publishing. https://doi.org/10.1007/978-3-319-13359-1_5

Zhang, H., Li, H., & Tam, C. (2006). Particle swarm optimization for resource-constrained project scheduling. *International Journal of Project Management*, *24*(1), 83-92. https://doi.org/10.1016/j.ijproman.2005.06.006

Zhang, H., Li, X., Li, H., & Huang, F. (2005). Particle swarm optimization-based schemes for resource-constrained project scheduling. *Automation in Construction*, *14*(3), 393-404. https://doi.org/10.1016/j.autcon.2004.08.006

Zhang, J., & Ding, X. (2011). A Multi-Swarm Self-Adaptive and Cooperative Particle Swarm Optimization. *Engineering Applications of Artificial Intelligence*, *24*(6), 958-967. https://doi.org/10.1016/j.engappai.2011.05.010

Zhong, L., & Lei, H. (2010, 20-22 Aug. 2010). *A mixed discrete particle swarm optimization for TSP*. IEEE International Conference on Advanced Computer Theory and Engineering, Chengdu, China. IEEE. https://doi.org/10.1109/ICACTE.2010.5579238

Ziari, I., Ledwich, G., & Ghosh, A. (2011). Optimal integrated planning of MV–LV distribution systems using DPSO. *Electric Power Systems Research*, *81*(10), 1905-1914. https://doi.org/10.1016/j.epsr.2011.05.015