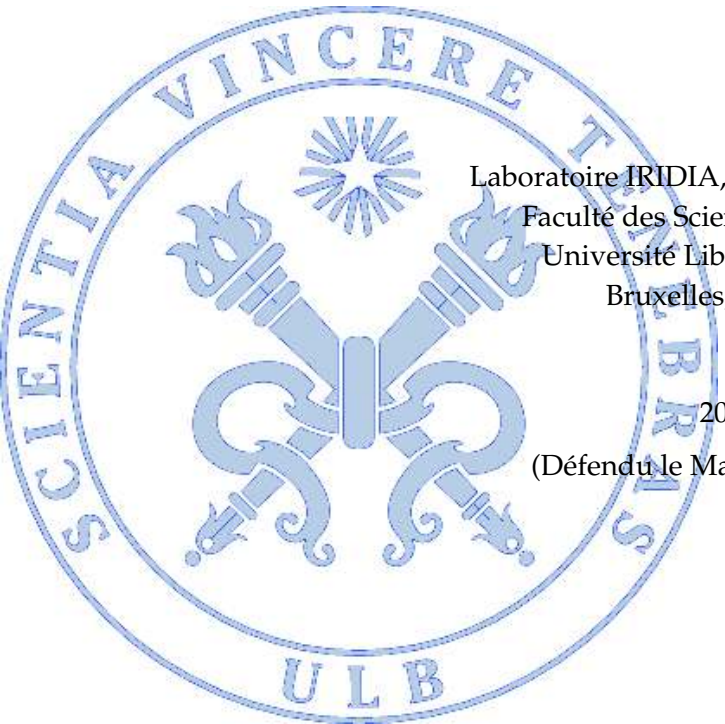
L'heuristique de la Gestalt : une méta-modélisation dynamique en ligne comme assistance du processus d'une métaheuristique

Thèse présentée par
Christophe Philemotte

En vue de l'obtention du grade de Docteur en Sciences de l'Ingénieur
Sous la direction de Hugues Bersini



Laboratoire IRIDIA, Département CoDE
Faculté des Sciences Appliquées
Université Libre de Bruxelles
Bruxelles, Belgique

2009

(Défendu le Mardi 9 Juin 2009)

© 2009
Christophe Philemotte
All Rights Reserved

Jury de thèse

Président :

Pr. Esteban Zimanyi,
I&R - CoDE, Université Libre de Bruxelles, Belgique

Secrétaire :

Dr. Mauro Birattari,
IRIDIA - CoDE, Université Libre de Bruxelles, Belgique

Promoteur :

Pr. Hugues Bersini,
IRIDIA - CoDE, Université Libre de Bruxelles, Belgique

Membres internes :

Dr. Marco Dorigo,
IRIDIA - CoDE, Université Libre de Bruxelles, Belgique

Pr. Tom Lenaerts,
MLG - DI, Université Libre de Bruxelles, Belgique

Dr. Thomas Stützle,
IRIDIA - CoDE, Université Libre de Bruxelles, Belgique

Membres externes :

Pr. Mark Bedau,
Reed College, Etats-Unis

Pour Odile et Mara . . .

Résumé

De nos jours, il est peu de processus ou de tâches qui ne requièrent pas l'optimisation d'une quantité : diminuer le temps de livraison, diminuer l'espace utilisé, réduire les efforts de développement, ... C'est donc sans surprise que la recherche en optimisation soit l'un des domaines les plus actifs des sciences des technologies de l'information. En optimisation combinatoire, les métaheuristiques sont à compter parmi le fleuron des techniques algorithmiques. Mais ce succès est encore au prix d'une quantité significative de temps de conception et développement. Ne serait-il pas possible d'aller encore plus loin ? D'automatiser la préparation des métaheuristiques ? En particulier dans des conditions telles le manque de temps, l'ignorance de techniques spécialisées ou encore la mauvaise compréhension du problème traité ? C'est ce à quoi nous répondons dans la présente thèse au moyen d'une approche de méta-modélisation de la recherche : l'heuristique de la Gestalt.

Considérant la représentation du problème comme un levier que l'on peut activer sous le processus de recherche mené par une métaheuristique, la thèse suggère la construction d'une abstraction de cette représentation capable d'assister la métaheuristique à trouver de bonnes solutions en contraignant sa recherche. Cette approche, inspirée de la psychologie de la Gestalt, nous l'appelons l'heuristique de la Gestalt. Son fonctionnement repose principalement sur l'agrégation des variables de la représentation. Cette agrégation donne lieu à une abstraction structurelle, mais également fonctionnelle en ce sens que les opérateurs de la métaheuristique doivent désormais respecter l'intégrité des agrégats définis.

Après avoir établi le contexte de la dissertation, nous discutons de la transposition de la psychologie de la Gestalt dans le cadre de l'optimisation combinatoire et des métaheuristiques. S'ensuit la formalisation de l'heuristique de la Gestalt et la description de sa réalisation. Finalement, une série d'études expérimentales sont menées pour éprouver le concept avancé et valider l'implémentation basée sur les algorithmes évolutionnistes que nous proposons. En conclusion, nous affirmons que l'implémentation de l'heuristique de la Gestalt basée, entre autres, sur un algorithme génétique de groupement est capable d'assister positivement des algorithmes génétiques lorsque les instances de problèmes traitées possèdent une structure riche et complexe, que leur taille est importante, que l'on est tôt dans le processus d'optimisation et que l'algorithme génétique n'est pas paramétré spécifiquement.

Mots clefs : optimisation combinatoire, métaheuristique, heuristique, représentation, Gestalt, prototypage rapide, méta-modélisation, abstraction.

Table des matières

Résumé vii

Table des matières xiv

Liste des Abréviations et Symboles xvii

1 Introduction 1

1.1 Objectif . . . 3
1.2 La psychologie de la Gestalt . . . 5
1.3 Contributions et assertions . . . 8
1.4 Publications scientifiques . . . 11
1.5 Plan de la thèse . . . 13

2 Contexte 15

2.1 Optimisation . . . 16
2.1.1 Problème d'optimisation . . . 16
2.1.2 Problème d'optimisation combinatoire . . . 17
2.1.2.1 Problème du voyageur de commerce . . . 17
2.1.2.2 Problème du maximum de un . . . 17
2.1.2.3 Représentation . . . 18
2.1.3 Complexité d'un problème . . . 19

2.2 Méthode algorithmique approximative pour l'optimisation combinatoire . . . 20
2.2.1 Métaheuristique . . . 20
2.2.2 Représentation et métaheuristique . . . 22
2.2.3 Graphe de voisinage . . . 23
2.2.4 Algorithme évolutionniste . . . 23
2.2.4.1 Fonction d'adaptation . . . 25
2.2.4.2 Représentation génétique . . . 25
2.2.4.3 Opérateurs évolutionnistes . . . 26
2.2.5 Paysage et structure . . . 27
2.3 Conclusion . . . 27

3 Une Gestalt en optimisation combinatoire 29
3.1 La psychologie de la Gestalt . . . 30
3.1.1 Les lois de l'organisation de la perception . . . 31
3.1.2 La Gestalt appliquée à la résolution de problème . . . 31
3.2 Une Gestalt pour les métaheuristiques . . . 33
3.2.1 Représentation comme perception du problème . . . 34
3.2.2 Une Gestalt canonique : l'agrégation de variables . . . 35
3.2.3 Une Gestalt comme abstraction de la recherche . . . 35
3.2.4 Construction d'une Gestalt . . . 37
3.2.4.1 Investigation . . . 37
3.2.4.2 Perturbation . . . 37
3.2.4.3 Contextualisation . . . 38
3.3 Une méta-modélisation dynamique et en ligne . . . 38
3.4 Conclusion . . . 39

4 L'heuristique de la Gestalt 41
4.1 Formalisation . . . 42
4.1.1 Abstraction structurelle : Gestalt description . . . 42
4.1.2 Abstraction fonctionnelle : lentille . . . 44
4.1.3 L'heuristique de la Gestalt . . . 47
4.2 Application d'une lentille à un algorithme évolutionniste . . . 50
4.2.1 Représentation . . . 51

4.2.2 Evaluation 51
4.2.3 Opérateurs 52
4.2.3.1 Mutation 52
4.2.3.2 Recombinaison 52
4.2.4 Conclusion 53
4.3 Implémentation évolutionniste de l'heuristique de la Gestalt 54
4.3.1 Un AG de groupement pour évoluer les lentilles . . . 55
4.3.2 Représentation des lentilles 56
4.3.3 Opérateurs évolutionnistes 57
4.3.3.1 Mutation 57
4.3.3.2 Inversion 57
4.3.3.3 Recombinaison 57
4.3.4 Structure de population 58
4.3.5 Evaluation des lentilles 58
4.3.6 Sélection des lentilles 60
4.3.7 Migration des individus 61
4.3.8 L'algorithme complet 61
4.4 Conclusion 61

5 Expériences : études et résultats 65
5.1 Quelques considérations préliminaires 66
5.1.1 Paramétrage des algorithmes utilisés 66
5.1.2 Mesure du temps et critère d'arrêt 67
5.1.3 Mesure d'efficacité 67
5.2 Une lentille statique et conçue a priori 68
5.2.1 Challenge de la Route Royale 68
5.2.2 Simple Algorithme Génétique et son paramétrage . . 70
5.2.3 La conception a priori des lentilles 71
5.2.4 Conditions expérimentales 72
5.2.5 Résultats et discussion 72
5.2.6 Conclusion 74
5.3 Une version biaisée et simplifiée de l'heuristique de la Gestalt 75
5.3.1 Problème du Si-et-seulement-Si Hiérarchique 76

5.3.2 Simple Algorithme Génétique et son paramétrage . . 78
5.3.3 Simplification et spécialisation de l'heuristique de la Gestalt 79
5.3.3.1 Spécialisation des lentilles au SISHI 79
5.3.3.2 Evolution des lentilles 83
5.3.4 Conditions expérimentales 85
5.3.5 Résultats et discussion 86
5.3.6 Conclusion 90
5.4 Etude de l'implémentation de l'heuristique de la Gestalt . . 91
5.4.1 Problèmes traités : Problème du Maximum de Un et du Si-et-seulement-Si Hiérarchique, deuxième version 92
5.4.1.1 Problème du Maximum de Un 92
5.4.1.2 Deuxième version du problème du Si-et-seulement-Si Hiérarchique Mélangé 93
5.4.2 Simple Algorithme Génétique et son paramétrage . . 94
5.4.3 L'heuristique de la Gestalt et son paramétrage 96
5.4.4 Conditions expérimentales 98
5.4.5 Résultats et discussion 99
5.4.5.1 Problème de Maximum de Un 99
5.4.5.2 Deuxième version du problème du Si-et-seulement-Si Hiérarchique Mélangé 112
5.4.6 Conclusion 125
5.5 Autre problème, autre représentation, autre Algorithme Génétique 127
5.5.1 Problème du Voyageur de Commerce 127
5.5.2 Algorithme Génétique d'Ordonnancement et son paramétrage 129
5.5.2.1 Représentation du Problème du Voyageur de Commerce 129
5.5.2.2 Opérateurs génétiques de croisement 130
5.5.2.3 Paramétrage 130
5.5.3 L'heuristique de la Gestalt : application pratique et paramétrage 131
5.5.3.1 Permutation, Gestalt description et lentille . 132

5.5.3.2 Introduction d'un biais positif quant au Problème du Voyageur de Commerce Uniforme 133
5.5.3.3 Paramétrage 133
5.5.4 Conditions expérimentales 135
5.5.5 Résultats et discussion 135
5.5.6 Conclusion 140
5.6 Conclusion 141

6 Travaux apparentés **145**
6.1 Algorithmes de recherche à base de modèle 145
6.2 Raffinement multi-niveaux pour problème d'optimisation combinatoire 146
6.3 Recherche à voisinage variable 147
6.4 Algorithmes évolutionnistes 148
6.4.1 Schemata 149
6.4.2 Hypothèse des blocs constitutifs et apprentissage de lien 149
6.4.3 Evolution compositionnelle 150
6.5 Hyperheuristique 151
6.6 Emergence intrinsèque 152
6.7 Conclusion 153

7 Conclusion **155**
7.1 Assister d'autres métaheuristiques 158
7.2 Implémentation de l'heuristique de la Gestalt 159
7.3 Une Gestalt plus sophistiquée 160

A Simple Algorithme Génétique **161**
A.1 Population et individu 161
A.2 Initialisation 161
A.3 Sélection 162
A.4 Recombinaison 162
A.5 Mutation 163

B Algorithme Génétique de Groupement **165**

B.1 Représentation 165
B.2 Recombinaison 166
B.3 Mutation 167
B.3.1 Redéfinition de l'appartenance d'un objet 167
B.3.2 Division d'un groupe 167
B.4 Fusion de deux groupes 168
B.5 Inversion 168

C Algorithme Génétique d'Ordonnancement **169**
C.1 Représentation 169
C.2 Recombinaison 170
C.2.1 Recombinaison par Correspondance Partielle 170
C.2.2 Croisement par Recombinaison d'Arêtes 171
C.3 Mutation par échange de loci adjacents 172
C.4 Sélection par tournoi 172

D Modèle Orienté Objet de la Gestalt **173**

Bibliographie **184**

Liste des Abréviations et Symboles

Abréviations

AE Algorithme Evolutionniste

AG Algorithme Génétique

AGG Algorithme Génétique de Groupement

AGO Algorithme Génétique d'Ordonnancement

CRA Croisement par Recombinaison d'Arêtes

CRR Challenge de la Route Royale

EUS Echantillonnage Universel Stochastique

NP temps Non-déterministe Polynomial

PMU Problème de Maximum de Un

POC Problème d'Optimisation Combinatoire

P temps Polynomial

PVC Problème du Voyageur de Commerce

PVC GR Problème du Voyageur de Commerce de type en GRappe

PVC UN Problème du Voyageur de Commerce de type UNiforme

R1P Recombinaison à Un Point

RCP Recombinaison par Correspondence Partielle

SAG Simple Algorithme Génétique

SISHI SI-et-seulement-Si HIérarchique

SISHIM SISHI Mélangé

Symboles

Δf erreur relative par rapport à l'optimum de la fonction objective f

$\mathbb{D}_x$ domaine associé à la variable x

$|E|$ taille de l'ensemble E

f fonction objective

$\hat{f}$ fonction d'adaptation ou d'évaluation

G ensemble des Gestalt variables ou Gestalt description

g Gestalt variable ; ou fonction d'évaluation d'une lentille

γ Gestalt de représentation

$\mathcal{G}$ lentille

N relation de voisinage

n taille de l'ensemble X, ou d'une instance ; ou fonction de voisinage

N_{eli} part d'élites dans une population

N_{pop} taille de population

ν graphe de voisinage

P population

φ représentation

Π un problème

π une instance du problème Π

P_I probabilité d'inversion

$P_{M\mu}$ probabilité de mutation (fusion)

P_μ probabilité de mutation

$P_{R\mu}$ probabilité de mutation (ré-assignation)

$P_{S\mu}$ probabilité de mutation (séparation)

P_X probabilité de recombinaison

r nombre de lentilles et sous-populations

ρ taille d'une sous-population

ρ_m taille de migration

S	espace de recherche
s	solution candidate
s^*	optimum global
t	temps discret ou continu
t_m	fréquence de migration
$\mathcal{V}$	ensemble des valuations
X	ensemble des variables ou description
x	variable ; ou n-uplet de variables

1 Introduction

En ce début de XXI^{e} siècle, il est une tâche particulièrement importante à automatiser et entrant à divers niveaux de conception, de production, voire de fonctionnement : l'*optimisation*, tâche ayant pour objectif de rechercher le moyen de modifier un sujet de telle sorte à en retirer un bénéfice. Voici quelques exemples de problèmes d'optimisation : augmenter la quantité de vaisselle dans un lave-vaisselle en la rangeant au mieux, réduire la surface d'un circuit imprimé, augmenter le taux d'occupation d'auditoires d'université, ou encore augmenter la capacité de livraison d'une flotte de véhicules. L'optimisation de ces exemples, non représentatifs de tous les problèmes d'optimisation, constituent un genre particulier s'attachant à manipuler un nombre fini d'éléments afin d'en construire une certaine configuration rapportant le plus de bénéfice : l'*optimisation combinatoire* (Cook et al., 1997). Les problèmes de ce type, nous en rencontrons tous les jours, même dans nos usages quotidiens.

Il est un moment où le nombre d'éléments à manipuler est tel que le nombre de possibles solutions explose. Par exemple, si le problème revient à constituer un arrangement [1] avec répétition de n objets pris k à k, le nombre de possibilités s'élève à n^k. Aussi, ce nombre de possibilités augmente-t-il exponentiellement avec le nombre d'objets à manipuler : c'est l'*explosion combinatoire*. Automatiser l'optimisation comme une simple énumération de toutes les possibilités devient alors très vite irréaliste, et ce quel que soit la puissance ou le nombre d'ordinateurs mis en jeu. C'est pourquoi des scientifiques et des ingénieurs élaborèrent et élaborent encore aujourd'hui des

1. Un arrangement correspond à un rangement de k objets choisis parmi n objets discernables, chaque objet pouvant être répété.

algorithmes d'optimisation efficaces, autrement dit des moyens d'automatiser efficacement l'optimisation. Pendant un temps, certains optimistes crurent que pour tout problème, on trouverait un algorithme efficace. Quand bien même nous serions optimistes, nous ne pouvons plus, à ce jour, nous bercer de telles illusions. Des théoriciens de l'informatique ont en effet démontré qu'il existait une famille de problèmes, que l'on pourrait qualifier de *difficiles*, pour lesquels nous ne connaissions pas, et ne connaissons toujours pas, d'algorithme efficace, et ont conjecturé la possibilité qu'il ne soit pas possible d'en concevoir (Papadimitriou, 1994).

Pour y remédier, diverses approches ont alors été élaborées par les scientifiques et les ingénieurs. Entre autres, il est une approche qui tend à se généraliser ces dernières années : les *méthodes algorithmiques approximatives* suggèrent, non plus de trouver la solution exacte, mais une *bonne solution* en un temps raisonnable (Blum and Roli, 2003). Parmi ces méthodes, les *métaheuristiques*, des stratégies *paramétrables* de haut niveau couplant différentes procédures appelées *opérateurs*, se sont distinguées par leur malléabilité, leur généralité et leur efficacité, et sont désormais à compter parmi l'état de l'art (Hoos and Stützle, 2004). Beaucoup de ces métaheuristiques ont été inspirées par des phénomènes naturels, comme par exemple la théorie de l'évolution qui permit d'élaborer les *algorithmes génétiques* (Holland, 1992). Bien qu'ayant pour objectif d'être détachées d'un problème particulier, les métaheuristiques nécessitent certaines attentions afin de développer tous leurs potentiels et de se positionner comme un fleuron de l'état de l'art. Aussi, compte tenu d'un problème donné, il convient de soigneusement choisir les opérateurs, les paramètres de configuration et la manière dont les possibles solutions sont *représentées*. L'efficacité est au prix d'un temps supplémentaire de développement.

Cette *spécialisation* n'est bien entendu pas toujours possible. Parfois, il n'est nullement possible d'effectuer un choix objectif : le problème est que trop peu compris, il est tout simplement mal défini, ou on ne connaît pas encore de moyen d'exploiter la connaissance à son propos. Aucune connaissance ne peut être alors mise à profit, et on ne peut que laisser la métaheuristique se débrouiller avec leurs solutions génériques. Ce scénario est celui dit de la *boîte noire* : le problème est vu comme une boîte noire dont rien ne transpire si ce n'est l'évaluation d'une possible solution. L'ignorance n'est pas le seul cas de figure donnant lieu à un tel scénario. Par exemple, on peut y compter le manque cruel de ressources nécessaires au développement et le souhait d'un prototypage rapide. Vient alors la question légitime : pouvons-nous y remédier ?

1.1 Objectif

Nous venons d'esquisser le contexte dans lequel s'inscrira la thèse que nous défendons dans le présent ouvrage. Ce contexte, nous pouvons en dégager la liste synthétique suivante :

- l'optimisation combinatoire,
- les métaheuristiques, en particulier les algorithmes génétiques, et
- le scénario de la boîte noire incluant les cas de figures :
  - d'une mauvaise connaissance du problème,
  - des solutions opératoires ad hoc inconnues,
  - des ressources nécessaires au développement maigres, ou
  - d'un prototypage rapide.

Et partant de là, avons-nous conclu en nous demandant s'il était possible d'y remédier. Cette question, nous proposons de la poursuivre dans la présente thèse.

Précisons plus avant cet objectif. Lors de son application à un problème, nous l'avons déjà évoqué, la métaheuristique nécessite certaines attentions à propos des paramètres de sa configuration, des opérateurs et de la représentation des solutions. L'efficacité obtenue dépend grandement de cette préparation. Aussi, nous chercherons au cours de cette thèse le moyen d'automatiser et d'intégrer une partie de cette préparation dans le contexte décrit ci-avant. Toutefois, il convient d'énoncer les solutions existantes se concentrant sur l'automatisation d'une partie de la préparation : le paramétrage automatique, la sélection automatique d'opérateurs, et l'adaptation automatique de la représentation.

La plupart des métaheuristiques nécessitent leur configuration via un ensemble de paramètres (Bartz-Beielstein, 2006; Birattari, 2004; Blum and Roli, 2003; Hoos and Stützle, 2004). La sélection de paramètres adéquats permet alors d'adapter la métaheuristique à un problème particulier, et d'atteindre des niveaux de performances compétitifs. Des algorithmes ont été développés afin d'automatiser ce paramétrage hors ou en ligne du processus d'optimisation, tel que l'agorithme F-Race (Birattari, 2004; Birattari et al., 2002), le "Sequential parameter optimization" (Bartz-Beielstein, 2006), ou de manière plus générale la famille des algorithmes d'optimisation à recherche réactive (Battiti et al., 2008).

Le fonctionnement des métaheuristiques repose sur des procédures unitaires, les opérateurs. La sélection soignée et le développement ad hoc de ceux-ci conditionnent, au même titre que le paramétrage, les performances d'une métaheuristique traitant un certain problème. L'automatisation de cette tâche, notamment en faisant appel à des techniques d'apprentissage, a donné naissance à une nouvelle famille de méthodes : les hyperheuristiques. Elles adaptent ou sélectionnent des opérateurs de telle manière à

en obtenir la meilleure combinaison au sein d'une métaheuristique (Burke et al., 2003; Cowling et al., 2001; Ross, 2005).

Les solutions possibles sont représentées d'une certaine manière. Cette manière conditionne pleinement les manipulations possibles que peuvent effectuer les opérateurs d'une métaheuristique. Communément, lorsque le problème est bien compris, la représentation dérive directement du modèle mathématique du problème. Les opérateurs sont alors développés en fonction de cette représentation. Aussi, pour la plupart des métaheuristiques, la représentation est posée a priori et n'est donc pas vue comme une problématique en soi. Il est pourtant une famille de métaheuristiques fort généralistes où la représentation est systématiquement prise en considération : les algorithmes évolutionnistes, dont font partie notamment les algorithmes génétiques. Au sein de la communauté se réunissant autour de ces algorithmes, un petit nombre d'approches d'adaptation simple de représentation spécifique ont été élaborées (Barbulescu et al., 2000; Mathias and Whitley, 1994; Rana and Whitley, 1997; Skolicki and Jong, 2004). Prometteuses dans le principe, ces approches se cantonnent aux algorithmes évolutionnistes et n'ont jamais été étendues à toute métaheuristique.

Plusieurs solutions ont ainsi été élaborées d'automatiser les tâches conditionnant la métaheuristique dans sa meilleure forme. Certaines proposent même de l'effectuer en ligne du processus d'optimisation. Et de nombreuses méthodes de ce genre restent certainement à développer, tout comme cette thèse en développe une.

Mais avant d'introduire plus avant la solution développée au sein de la présente thèse, rappelons précisément les termes de l'objectif poursuivi. Nous cherchons à :

> *élaborer une méthode algorithmique permettant d'assister en ligne une métaheuristique dans sa tâche d'optimisation et ce dans les conditions proches du scénario de la boîte noire.*

Par assistance, nous entendons une amélioration de la qualité de la solution à temps constant ou une diminution du temps nécessaire à qualité constante de solution. Rappelons également que, dans le scénario de la boîte noire, nous englobons les cas de figure non-exclusifs suivants :

- la mauvaise connaissance du problème,
- l'ignorance de solutions opératoires ad hoc,
- l'absence de ressources nécessaires au développement, ou
- le prototypage rapide.

Nous avons délibérément formulé cet objectif de manière fort générale, espérant que notre travail ait à long terme un large impact. Cependant, nous nous limiterons dans la présente thèse à un cas plus concret :

> *élaborer une méthode algorithmique permettant d'assister en ligne un algorithme évolutionniste dans sa tâche d'optimisation au moyen*

d'une reformulation de la représentation des solutions, et ce dans les conditions proches du scénario de la boîte noire.

Et pour atteindre cet objectif, nous userons d'une stratégie mettant en jeu une reformulation de la représentation, stratégie librement inspirée d'une théorie générale de psychologie, comme nous allons l'expliquer dans la suite.

1.2 La psychologie de la Gestalt

La psychologie de la Gestalt fournit une théorie générale tentant de comprendre certains processus mentaux tels que la vision ou la résolution de problèmes comme le produit d'une systémique (Ash, 1995). Selon cette théorie, l'homme tend à considérer les perceptions dans leur ensemble structuré et non comme la simple addition de stimuli indépendants. Un holisme que l'on résume communément par la maxime : "le tout est plus que la somme des parties". De même, cette théorie s'applique à expliquer le processus cognitif de résolution de problème. Selon Wertheimer (1959), la résolution d'un problème n'est obtenu qu'au prix d'un embrassement de la structure totale du problème : une Gestalt, autrement dit comprendre, saisir ensemble, tous les composants du problème dans le contexte du dit problème.

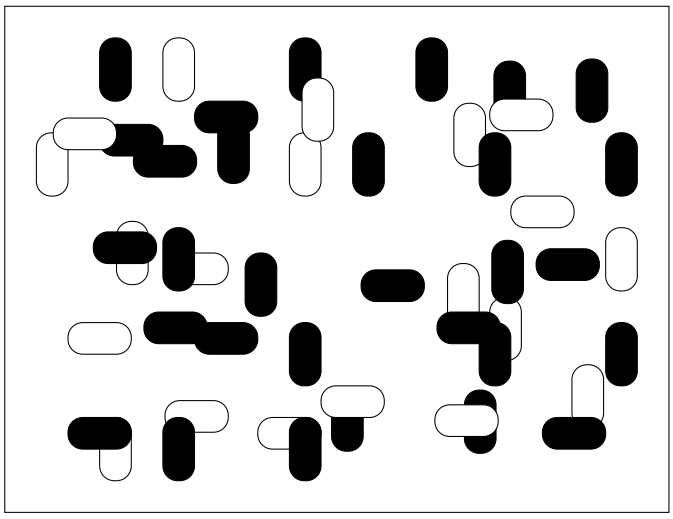


FIGURE 1.1 – Le problème du peintre : s'assurer que toutes les pièces soient blanches, dans le cas contraire, rectifier les pièces d'une autre couleur en blanc. La figure présente la disposition typique des pièces.

Afin d'illustrer quelque peu ces propos, imaginons un cas pratique. Un peintre a reçu pour tâche de s'assurer que toutes les pièces dans une boîte soit blanches. Si certaines pièces ne revêtissent pas cette couleur, le peintre

doit alors les peindre en blanc. Malheureusement pour le peintre, les pièces sont totalement désordonnées dans la boîte, comme par exemple sur la Fig. 1.1. La première fois, le peintre se contente de repérer individuellement les pièces noires et de les peindre une à une. Les pièces se chevauchant de manière totalement désordonnées, le peintre doit rester concentrer pour identifier les pièces erronées pour pouvoir accomplir sa tâche. Il perd dès lors un temps précieux l'empêchant d'être pleinement efficace.

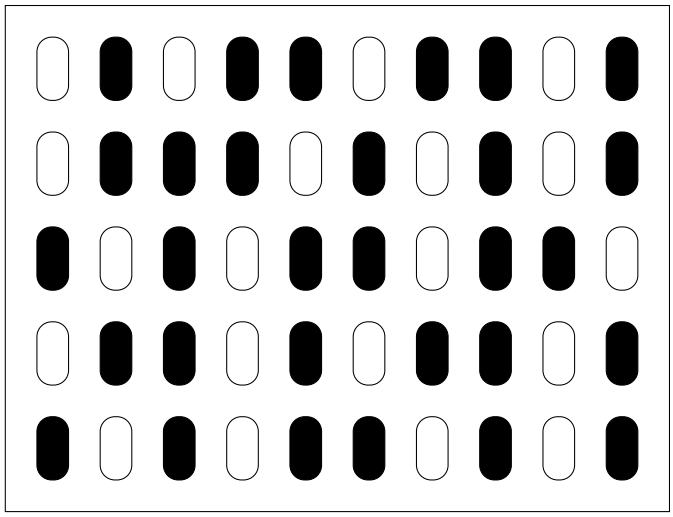


FIGURE 1.2 – Le problème du peintre : s'assurer que toutes les pièces soient blanches, dans le cas contraire, rectifier les pièces d'une autre couleur en blanc. La figure présente la disposition des pièces après un rangement quadrillé.

Notre peintre n'étant pas un imbécile, il décide de réarranger les pièces dans la boîte avant d'effectuer la tâche qui lui a été assigné. Partant du coin en haut à gauche de la boîte, chaque fois qu'il rencontre une pièce, il la range selon une grille imaginaire telle celle présentée sur la Fig. 1.2. Le rangement une fois effectué, le peintre se décide alors à parcourir cette grille de pièces pour identifier les erreurs et les rectifier. Ayant doté la boîte d'une structure, il parcourt la grille ligne par ligne, pièce par pièce pour vérifier la couleur, et le cas échéant la modifier. Cette fois, le travail est efficace, et le peintre n'est plus dérangé inutilement par le désordre initial.

Mais notre peintre n'est toujours pas satisfait. Il sait qu'il pourrait faire mieux. Amateur de la psychologie de la Gestalt, il se décide à appliquer l'un de ces principes et de créer une Gestalt, c'est-à-dire une perception d'ensemble. Pour ce faire, il range les pièces selon leur couleur. Aussi, chaque fois qu'il range une pièce, tient-il compte de sa couleur pour la disposer dans sa grille de rangement tel que l'illustre la Fig. 1.3 : les blanches au-dessus, les autres en-dessous . Ordonnées de la sorte, les pièces non blanches se retrouvent toutes les unes à côté des autres. Et d'un seul coup d'oeil, le

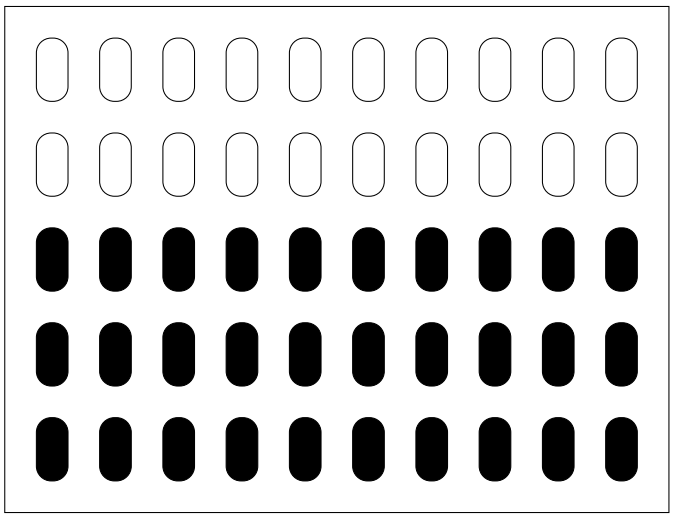


FIGURE 1.3 – Le problème du peintre : s'assurer que toutes les pièces soient blanches, dans le cas contraire, rectifier les pièces d'une autre couleur en blanc. La figure présente la disposition des pièces après un rangement quadrillé selon la couleur.

peintre identifie toutes celles qui nécessitent un bon coup de pinceau, et les peint d'un seul coup de pinceau généreux. Cette fois, le peintre estime qu'il est dès lors bien plus efficace. Certes il perd un peu de temps à ranger les pièces, mais le gain de temps obtenu justifie amplement cette perte de temps.

De ce petit exemple pratique, nous pouvons retirer deux leçons importantes. Premièrement, la représentation du problème influence fortement l'efficacité avec laquelle on peut le résoudre. Deuxièmement, la manière dont on peut résoudre le problème dépend de la représentation. Et ces leçons nous pouvons les transposer à l'optimisation combinatoire. Ce petit exemple peut d'ailleurs être interprété selon cette perspective : le nombre de pièces blanches doit être maximisé, et pour ce faire nous pouvons agir en changeant la couleur d'une pièce. Par analogie, nous pouvons substituer :

- le peintre par une métaheuristique,
- l'état de toutes les pièce par une solution possible,
- la manière dont sont rangées les pièces par une représentation, et
- la manière dont le peintre peint les pièces par un opérateur.

Et voilà que la psychologie de la Gestalt nous inspire une méthode algorithmique pour assister une métaheuristique : la construction d'une Gestalt de la représentation du problème. Cette approche rejoint celles dédiées à l'adaptation de la représentation sans pour autant se restreindre à une seule famille particulière de métaheuristique. De plus, elle dépasse la simple adaptation en modifiant également la manière dont les opérateurs agissent,

la rapprochant de ce fait des hyperheuristiques. Portant sur la représentation et les opérateurs, nous sommes en présence d'une nouvelle approche.

Plus précisément, la Gestalt modifie globalement la manière dont une métaheuristique procède, elle modifie son modèle de recherche. En somme, la Gestalt se constitue comme un méta-modèle de recherche, et la construction d'une Gestalt comme une méta-modélisation. Telle est la méthode algorithmique d'assistance que nous développons et défendons au sein de cette thèse. Dès lors, l'objectif, faisant le sujet de cette thèse, peut être exprimé en ces termes :

> *élaborer une méta-modélisation dynamique inspirée de la psychologie de la Gestalt permettant d'assister en ligne un algorithme évolutionniste dans sa tâche d'optimisation et ce dans les conditions proches du scénario de la boîte noire.*

Répondant à ce but, nous proposons l'heuristique de la Gestalt. Elle fait suite à une série de contributions que nous allons énumérer et présenter brièvement dans ce qui suit.

1.3 Contributions et assertions

L'élaboration d'une méta-modélisation inspirée de la psychologie de la Gestalt s'est déclinée au travers d'un certain nombre d'étapes dirigées par diverses questions. Ces questions, nous allons les exposer ci-après en précisant à chaque fois les contributions et les assertions que nous avons pu y donner.

Comment formalise-t-on la représentation, les opérateurs et leur conjonction dans le cadre de l'optimisation combinatoire et des métaheuristiques ? Après reprise et uniformisation de divers formalismes existants, nous fournissons un cadre permettant de définir avec rigueur ce que nous entendons par représentation, opérateur et leur combinaison, induisant le graphe de voisinage dans le cadre de l'optimisation combinatoire et des métaheuristiques. Ce cadre ne se limite dès lors plus aux seuls algorithmes évolutionnistes et nous permettra de définir formellement et en toute généralité l'heuristique de la Gestalt.

Comment la psychologie de la Gestalt est-elle exactement transposée à l'optimisation combinatoire et aux métaheuristiques, et en quoi consiste-t-elle ? Inspirés par la psychologie de la Gestalt, nous développons un concept analogue appliqué aux métaheuristiques et l'optimisation combinatoire : la Gestalt de représentation. Il s'agit de former des agrégats d'éléments constitutifs de la représentation. Ces agrégats insécables deviennent les nouvelles fondations sur lesquelles la représentation est reformulée. De ce fait, les opérateurs doivent les considérer en tant que tels et sont modifiés conséquemment. Se

fondant principalement sur la combinaison de ces dits opérateurs, le processus de recherche de la métaheuristique est alors altéré. Mais une Gestalt ne peut être fixe par essence, aussi pouvons-nous également transposer les principes spécifiquement établis afin d'expliquer les processus cognitifs de résolution de problème : investigation, perturbation et contextualisation. Et nous voilà avec un fil conducteur pour concevoir l'heuristique de la Gestalt, autrement dit un mécanisme capable d'apprendre, de construire et d'adapter une Gestalt de représentation.

A quel type de problèmes l'heuristique de la Gestalt convient-elle ? Lato sensu, une Gestalt de représentation est l'expression d'une structure particulière. Aussi, il est attendu que l'heuristique de la Gestalt assiste l'algorithme d'optimisation lorsque le problème traité arbore effectivement certaines structures. Sans cela, elle ne peut mettre en exergue quoi que ce soit. Cette hypothèse, nous avons pu la vérifier expérimentalement.

Comment formalise-t-on l'heuristique de la Gestalt le plus généralement possible ? Nous basant sur le cadre formel définissant la représentation, les opérateurs et leur conjonction, nous formalisons l'heuristique de la Gestalt au travers de trois volets. Le premier permet de définir la Gestalt de représentation en tant que telle. Le deuxième concerne l'opérateur mathématique résultant, appelé lentille, qui assure l'intégrité de la Gestalt définie lors d'une opération de métaheuristique. Et le troisième et dernier volet concerne le mécanisme d'apprentissage, de construction et d'adaptation de la Gestalt de représentation.

Comment applique-t-on des lentilles à des algorithmes évolutionnistes, en particulier un algorithme génétique ? Les lentilles sont les garantes de l'intégrité de la Gestalt de représentation et les représentantes de l'abstraction fonctionnelle induite. Sous l'influence d'une lentille, la métaheuristique qui est assistée voit se modifier son processus de recherche. Nous expliquons comment il est possible d'intégrer une lentille dans des algorithmes évolutionnistes, en particulier un simple algorithme génétique. L'ensemble est alors réalisé et testé.

Comment peut-on implémenter l'heuristique de la Gestalt au moyen d'algorithmes évolutionnistes ? La définition de l'heuristique de la Gestalt ne comporte aucune indication sur la manière de réaliser le mécanisme d'apprentissage et de construction. Aussi, nous proposons une possible implémentation basée sur un algorithme génétique de groupement, un modèle de multi-populations, un modèle de sélection multi-niveau et une co-évolution. De même, une série d'expériences sont menées afin d'étudier le rôle des différents composants algorithmiques de cette implémentation et de la valider dans des cas précis.

De ces diverses contributions, nous pouvons extraire plusieurs hypothèses. Afin d'y répondre, nous les avons éprouvées au moyen de quatre expériences ad hoc. S'ensuivent plusieurs assertions que nous pouvons résumer comme suit :

1. Conçues en respect de la structure d'une instance du challenge de la route royale, des lentilles statiques influencent positivement, significativement et pour un certain temps le processus de recherche mené par le simple algorithme génétique traitant une instance du challenge de la route royale.
2. Spécialisée pour le problème du si-et-seulement-si mélangé et implémentée au moyen d'un simple algorithme génétique, l'heuristique de la Gestalt assiste très positivement un simple algorithme génétique traitant diverses instances de diverses tailles de ce problème.
3. La construction des lentilles au moyen d'un algorithme génétique de groupement contribue positivement à l'assistance prodiguée à un simple algorithme génétique. Cette contribution est d'autant plus significative que le paramétrage du simple algorithme génétique assisté n'est pas spécifique aux instances de problème traitées, que la taille des instances est conséquente, que le problème considéré possède une structure riche et complexe comme la deuxième version du problème du si-et-seulement-si mélangé, et que l'on est tôt dans le processus d'optimisation.
4. La construction des lentilles au moyen d'un algorithme génétique de groupement montre des faiblesses à assister un simple algorithme génétique lorsque le paramétrage du simple algorithme génétique assisté est spécifique aux instances de problème traitées, que la taille des instances est petite, que le problème considéré est simple et linéaire comme le problème du maximum de un, et que l'on est tard dans le processus d'optimisation.
5. Légèrement spécialisée, l'heuristique de la Gestalt peut être appliquée avec succès à un autre type d'algorithmes génétiques comme celui d'ordonnancement pour traiter diverses instances du problème du voyageur de commerce. Elle apporte alors une meilleure accélération initiale de la qualité des solutions trouvées et une assistance s'améliorant relativement à la taille de l'instance considérée. Son interaction avec un opérateur générique telle la recombinaison à correspondance partielle est meilleure qu'avec un opérateur spécifique tel le croisement par recombinaison d'arêtes.

Un certain nombre de conclusions interprétatives à propos de l'heuristique de la Gestalt ont été tirées sur base de ces assertions et de leur confrontation avec les supputations conceptuelles :

- l'heuristique de la Gestalt peut se voir comme une excellente méthode d'initialisation de population de solutions,
- l'heuristique de la Gestalt s'avère un moyen générique et simple d'inoculer de la connaissance à propos du problème dans un algorithme génétique, et
- l'heuristique de la Gestalt se présente comme une bonne première méta-modélisation dynamique et en ligne pour l'assistance des algorithmes génétiques dans le cadre d'un scénario de la boîte noire.

Ceci nous permet de conclure ce bref parcours des contributions reportées dans cette thèse. Et de passer maintenant en revue les publications scientifiques effectuées dans ce cadre.

1.4 Publications scientifiques

La thèse que nous présentons est l'aboutissement d'un long développement initié fin de l'année 2004. Les différentes étapes de développement ont donné lieu aux publications scientifiques suivantes, ordonnées par ordre chronologique croissant :

(Philemotte and Bersini, 2005b) Christophe Philemotte and Hugues Bersini. Intrinsic emergence boosts adaptive capacity. In *GECCO '05 : Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 559–560. ACM Press, 2005.

(Philemotte and Bersini, 2005a) Christophe Philemotte and Hugues Bersini. Coevolution of effective observers and observed multi-agents system. In *Advances in Artificial Life, 8th European Conference, ECAL 2005, Canterbury, UK, September 5-9, 2005, Proceedings*, volume 3630 of *Lecture Notes in Computer Science*, pages 785–794. Springer, 2005.

(Philemotte, 2006) Christophe Philemotte. "Good" observers enhance SGA exploration. In Mario Giacobini and Jano van Hemert, editors, *First European Graduate Student Workshop on Evolutionary Computation*, pages 59–72, Budapest, Hungary, 2006.

(Philemotte and Bersini, 2006b) Christophe Philemotte and Hugues Bersini. How an optimal observer can collapse the search space. In *GECCO '06 : Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1273–1280. ACM Press, 2006.

(Philemotte and Bersini, 2006a) Christophe Philemotte and Hugues Bersini. How an optimal observer can smooth a landscape. In *IEEE Congress on Evolutionary Computation 2006*, pages 2293–2300. IEEE, 2006.

(Philemotte and Bersini, 2007) Christophe Philemotte and Hugues Bersini. A gestalt genetic algorithm : less details for better search. In Hod Lip-

son, editor, *GECCO '07 : Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1328–1334. ACM, 2007.

(Philemotte, 2008) Christophe Philemotte.A stochastic routing policy perspective for an operator-level hyperheuristic.Technical Report TR/IRIDIA/2008-018, IRIDIA, CoDE, Université Libre de Bruxelles, August 2008.

(Philemotte and Bersini, 2008) Christophe Philemotte and Hugues Bersini. The gestalt heuristic : learning the right level of abstraction to better search the optima. Technical Report TR/IRIDIA/2008-021, IRIDIA, CoDE, Université Libre de Bruxelles, October 2008.

Le développement de l'heuristique de la Gestalt fut initié sur base d'une idée de Bersini (2004). L'idée reposait sur l'utilité pragmatique de propriétés émergentes menant à l'élaboration d'un mécanisme capable de les repérer et de les utiliser. Partant de là, un premier algorithme fut développé de telle manière à automatiser le mécanisme préalablement proposé par Bersini (2004), et appliqué à un problème d'automates cellulaires (Philemotte and Bersini, 2005a,b). Dans un certain sens, cet algorithme renfermait les prémisses algorithmiques de l'heuristique de la Gestalt.

Replaçant ce premier algorithme dans un contexte plus large, nous en réalisâmes ensuite une première généralisation afin de pouvoir traiter tout type de problème d'optimisation combinatoire binaire (Philemotte, 2006; Philemotte and Bersini, 2006a). Nous avions là une forme simplifiée et spécialisée de l'heuristique de la Gestalt. Une version sur laquelle se base et s'inspire l'expérience que nous présenterons à la Sec. 5.3 du Chapitre 5.

Ces travaux nous fournirent le matériel nécessaire que pour interpréter le mécanisme que nous mettions en oeuvre. Et confronté à la psychologie de la Gestalt, nous trouvâmes une perspective d'ensemble menant au principe de méta-modélisation comme assistance du processus d'optimisation. Ces réflexions nous menèrent à la première version complète et avouée de l'heuristique de la Gestalt assistant, et à l'esquisse d'un formalisme général découplé d'une représentation spécifique. Les premières expériences s'effectuèrent sur l'algorithme génétique d'ordonnancement traitant le problème du voyageur de commerce (Philemotte and Bersini, 2007). Ces travaux expérimentaux ont inspiré l'expérience que nous présenterons à la Sec. 5.5 du Chapitre 5.

L'algorithme et l'implémentation ainsi obtenus étaient devenus assez matures que pour nous permettre de les formaliser de manière générale dans le cadre de l'optimisation combinatoire et les métaheuristiques. Ce formalisme une fois obtenu nous permît une meilleure compréhension de l'algorithme, notamment en effectuant un parallèle avec les hyperheuristiques (Philemotte, 2008). De plus, il permît d'asseoir sa structure algorithmique et sa réalisation, nous permettant de développer une méthodologie générique de son application. Et assistant des algorithmes génétiques, nous

pûmes l'éprouver expérimentalement dans divers cas de figure (Philemotte and Bersini, 2008). Ces derniers travaux ont d'une part permis de constituer le Chapitre 4 et, d'autre part, inspiré les expériences que nous présenterons aux Sec. 5.2, 5.4 et 5.5 du Chapitre 5. Ce bref passage en revue des publications scientifiques étant terminé, nous pouvons conclure ce chapitre d'introduction en décrivant la structure du présent document.

1.5 Plan de la thèse

Notre dissertation est organisée en six chapitres sans compter la présente introduction :

Chapitre 2 : Contexte Nous commençons par décrire le contexte dans lequel la thèse est développée. En particulier, nous établissons une définition formelle et uniforme de la représentation, des opérateurs et du graphe de voisinage. Ceci nous permettra par la suite de définir formellement l'heuristique de la Gestalt. Ajoutons que nous abordons également dans ce chapitre toutes les particularités propres aux algorithmes évolutionnistes puisque les algorithmes assistés et l'implémentation de l'heuristique de la Gestalt appartiennent à cette classe de métaheuristique.

Chapitre 3 : Une Gestalt en optimisation combinatoire Ensuite, nous introduisons la psychologie de la Gestalt et détaillons tous les principes qui nous permettront de construire son concept analogue en optimisation combinatoire. Ceci étant, nous attaquons la transposition de la Gestalt en optimisation combinatoire. In fine, nous obtenons le concept complet de l'heuristique de la Gestalt. Et nous concluons le chapitre en inscrivant cette démarche dans une approche plus vaste, la méta-modélisation dynamique et en ligne comme un moyen d'assistance.

Chapitre 4 : L'heuristique de la Gestalt Les différents concepts discutés au chapitre précédent nous amènent à définir formellement l'heuristique de la Gestalt sur base du formalisme présenté au Chapitre 2. La définition se structure en trois parties principales abordant dans l'ordre l'abstraction structurelle, l'abstraction fonctionnelle et l'heuristique de la Gestalt. La première partie permet la reformulation de la représentation originale en termes d'une Gestalt. La deuxième permet de transformer implicitement les actions des opérateurs d'une métaheuristique afin de respecter l'intégrité de la Gestalt appliquée. La dernière partie concerne le mécanisme permettant de construire par apprentissage la Gestalt. La formalisation étant définie, nous considérons comment l'heuristique de la Gestalt est interfacée avec une métaheuristique afin de l'assister. Pour ce faire, nous illustrons

cette démarche sur une exemple de métaheuristique, un algorithme génétique. Nous terminons ce chapitre en abordant en détails l'implémentation évolutionniste de l'heuristique de la Gestalt que nous proposons. Cette implémentation repose sur l'utilisation d'un algorithme génétique de groupement, d'un modèle de multi-populations, d'un modèle de sélection multi-niveaux et d'une coévolution.

Chapitre 5 : Expériences : études et résultats Vient alors le temps d'étudier et d'éprouver l'heuristique de la Gestalt. Quatre expériences sont présentées à diverses fins. La première expérience est une preuve du concept de l'impact effectif d'une Gestalt de représentation sur l'optimisation menée par un simple algorithme génétique. La deuxième expérience s'accorde à éprouver le concept d'un mécanisme capable d'élaborer une Gestalt de représentation spécifique au problème traité afin d'assister un simple algorithme génétique. Ces preuves de concept étant effectuées, la troisième expérience s'attelle à étudier l'implémentation de l'heuristique de la Gestalt que nous avons détaillée au Chapitre 4. En particulier, nous nous y assurons que la Gestalt de représentation et sa construction contribue positivement à l'assistance prodiguée à un simple algorithme génétique traitant des problèmes de diverses tailles et difficultés. Enfin, la dernière expérience se concentre sur une mise en situation pratique de l'heuristique de la Gestalt. L'expérience illustre ainsi la démarche ad hoc à adopter dans d'autres cas de figure : autre problème, autre représentation et autre algorithme génétique. Adoptant une attitude réaliste, nous montrons quelles modifications sont à opérer et les éventuelles spécialisations qui peuvent être introduites.

Chapitre 6 : Travaux apparentés Après une conséquente étude expérimentale, nous repositionnons l'heuristique de la Gestalt par rapport aux travaux qui pourraient s'y apparenter. C'est également l'occasion de lever toute ambiguïté à son propos.

Chapitre 7 : Conclusion Finalement, nous concluons la dissertation en la résumant, rappelant les observations expérimentales importantes et en discutant les travaux futurs qu'il serait bon de mener en conséquence.

2

Contexte

Le présent chapitre a pour objectif de dresser le contexte dans lequel la dissertation s'inscrira. Il est organisé en deux principales sections. La première de ces deux sections introduit le domaine de l'optimisation combinatoire et la terminologie appropriée. En particulier, nous y définissons ce que nous entendons par problème et représentation.

La seconde section se concentre quant à elle sur un type particulier d'algorithme d'optimisation combinatoire : les métaheuristiques dont nous dépeignons, dans un premier temps, un vaste panorama. Nous discutons ensuite du rôle de la représentation dans le fonctionnement des métaheuristique, et en particulier de sa conjonction avec les opérateurs. Cette dernière peut être modélisée comme le graphe de voisinage, le modèle de recherche d'une métaheuristique. Tout comme la représentation, nous prenons un soin particulier à le définir puisqu'il jouera un rôle important dans la suite de notre dissertation. Nous abordons ensuite une famille particulière de métaheuristiques que sont les algorithmes évolutionnistes. Nous y puiseront pour implémenter l'heuristique de la Gestalt, mais également pour choisir les algorithmes assistés par cette dernière. Nous clôturons cette deuxième section en précisant le sens du terme "structure", un terme que nous emploierons en plusieurs circonstances.

Le chapitre se termine par une simple conclusion rappelant les sujets importants pour la suite de la thèse. Mais dès à présent introduisons l'optimisation combinatoire.

2.1 Optimisation

S'il l'on se réfère à sa définition usuelle, l'optimisation consiste à améliorer. Autrement dit, il s'agit de trouver le moyen de modifier un sujet donné de telle manière à pouvoir en retirer un bénéfice quelconque. La signification se précise en mathématique : l'**optimisation** est l'étude du problème de l'extremum global s^* d'une fonction f d'un ensemble S dans un ensemble totalement ordonné, typiquement l'ensemble des nombres réels $\mathbb{R}$. Selon la fonction, cette étude peut être opérée analytiquement ou non. Dans ce dernier cas, il faut alors se rabattre sur des méthodes numériques afin de trouver l'extremum global. Plusieurs techniques de résolution ont ainsi été élaborées et peuvent, dans la plupart des cas, être automatisées sous forme d'algorithmes dont l'élaboration constitue un problème en soi en informatique. Aussi, le terme d'optimisation y réfère-t-il, par extension, au processus de recherche lui-même.

2.1.1 Problème d'optimisation

Une **instance d'un problème d'optimisation** π peut être définie par une paire [1] (S, f), où S est l'ensemble des *solutions candidates* [2] et la fonction $f : s \to \mathbb{R}$ mesure la qualité d'une solution candidate. L'espace S est appelé l'*espace de recherche* et la fonction f la *fonction objective*.

Soit une instance donnée π, optimiser cette instance consiste à chercher la solution candidate $s^* \in S$ qui est un extremum global de f (Birattari, 2004; Hoos and Stützle, 2004). Plus précisément, selon que l'on souhaite minimiser ou maximiser la fonction f, l'extremum global s^* recherché sera :

- un minimum global tel que $f(s^*) \leq f(s)$, $\forall s \in S$, ou
- un maximum global tel que $f(s^*) \geq f(s)$, $\forall s \in S$.

Un problème d'optimisation peut être défini comme un ensemble Π d'instances π, qui sont généralement apparentées (Birattari, 2004; Hoos and Stützle, 2004). L'instance peut donc être vue comme un cas spécifique et concret d'un problème posé. En particulier, certains problèmes peuvent être définis par deux algorithmes $\mathbb{A}_S$ et $\mathbb{A}_f$ générant respectivement S et f selon un ensemble de paramètres donnés.

1. Une instance est parfois définie comme le triplet (S, f, Ω) où Ω est un ensemble de contraintes (Birattari, 2004; Hoos and Stützle, 2004). Respecter ces contraintes génère un sous-ensemble $S' \subseteq S$ de solutions admissibles et donc un nouvelle instance sous la forme de la paire (S', f). On peut donc en toute généralité se limiter à définir une instance d'un problème d'optimisation par la paire (S, f).
2. Une solution candidate est également appelée solution admissible.

2.1.2 Problème d'optimisation combinatoire

Une instance d'un **Problème d'Optimisation Combinatoire** (POC) est une instance d'un problème d'optimisation dont l'espace de recherche S est discret et fini (Blum and Roli, 2003; Cook et al., 1997; Hoos and Stützle, 2004; Papadimitriou and Steiglitz, 1998). Malgré la finitude de S, la cardinalité de S est en général d'un tel ordre que l'énumération des solutions candidates en est rendue impossible en un temps raisonnable. Ainsi, exprime-t-on les solutions candidates sous la forme d'une combinaison de composants. La génération de toutes les possibilités, pouvant en particulier s'obtenir par le biais d'un algorithme $\mathbb{A}_S$, permet alors de recréer individuellement les solutions candidates s de l'espace de recherche S.

Les POCs se révèlent être d'une grande importance pratique dans des domaines aussi variés que nombreux comme le problème du voyageur de commerce ou le problème de maximum de un.

2.1.2.1 Problème du voyageur de commerce

Le Problème du Voyageur de Commerce (PVC) consiste à trouver le plus court chemin passant par un ensemble de villes qui doivent toutes être visitées exactement une fois. Ce problème est l'un des plus étudiés et populaires, en partie du fait de sa simplicité d'énonciation et de compréhension, et de sa représentativité d'un large spectre d'applications.

Sa version symétrique peut être définie comme suit (Hoos and Stützle, 2004). Considérons un graphe simple, orienté et pondéré $G := (V, E, w)$, où V est l'ensemble des n sommets, $E \subseteq V \times V$ est l'ensemble des couples des sommets appelés arêtes et $w : E \to \mathbb{R}^+$ est une fonction associant à chaque arête $e := (v, v') \in E$ un nombre réel positif appelé poids $w(e)$ tel que $w((v, v')) = w((v', v))$, et où v, $v' \in V$. La tâche poursuivie consiste à rechercher le cycle hamiltonien[3] $p := (u_1, \ldots, u_n, u_1)$, où u_i est un sommet de V, minimisant le poids total du cycle p donné par la fonction réelle positive :

$$f_{\text{PVC}}(p) = w((u_n, u_1)) + \sum_{i=1}^{n-1} w((u_i, u_{i+1})) \,.$$

2.1.2.2 Problème du maximum de un

Le Problème du Maximum de Un (PMU) (Schaffer and Eshelman, 1991) est un simple problème de comptage de **1**s dans une chaîne de bits largement utilisé pour tester les performances d'un certain type d'algorithmes

3. Un cycle hamiltonien est un chemin cyclique qui visite chacun des sommet une et une seule fois, excepté pour son point de départ.

d'optimisation. Formellement, le problème consiste à trouver un n-uplet de bits $x = (x_1, \ldots, x_n)$, où $x_{i=1,\ldots,n} \in \{0,1\}$, qui maximise l'équation suivante :

$$f_{\text{PMU}}(x) = \sum_{i=1}^{n} x_i \ .$$

2.1.2.3 Représentation

Comme exprimé ci-avant, une solution candidate peut être exprimée sous la forme d'une combinaison de composants. Cette expression n'est autre que la **représentation** de la solution candidate, c'est-à-dire la manière dont il est possible d'exprimer une solution candidate en termes de composants. Dans l'idéalité de son rôle, la représentation, que l'on qualifiera de *complète*, exprimera uniquement toutes les solutions candidates et pourra, dans la majorité des cas, se dériver de l'un des modèles mathématiques du POC concerné.

Formellement [4], une représentation d'une instance π d'un POC se définit selon un ensemble de n variables $x \in X$, où $n \in \mathbb{N}_0$. Chaque variable $x \in X$ est associée à un ensemble de ses valeurs possibles $\mathbb{D}_x$, appelé domaine de x. La représentation d'une solution candidate s se définit comme étant un ensemble de n composantes canoniques où chaque composante $c(x)$ (ou c_x) est une valeur de $\mathbb{D}_x$ associée à la variable x. Remarquons que le nombre n définit implicitement la taille de l'instance. Un ensemble de n composantes c_x revient à définir une valuation. Une **valuation** est une application de X vers $\bigcup_{x \in X} \mathbb{D}_x$ telle que pour toute variable $x \in X$, il est associé une et une valeur $c(x) \in \mathbb{D}_x$. Une **représentation** complète de l'instance π est alors une bijection φ de S dans l'ensemble $\mathcal{V}$ des valuations. Cette bijection associe à toute solution candidate $s \in S$ une et une seule valuation $c \in \mathcal{V}$, ou représentation de s. Au final, une instance π d'un POC munie d'une représentation φ peut être définie comme le triplet $(X, \mathcal{V}, f)$.

Illustrons cette formalisation avec une représentation d'une instance du PMU, une instance d'une chaîne de trois bits. Canoniquement, une solution candidate est exprimée sous la forme d'un triplet (x_1, x_2, x_3), où les variables x_1, x_2 et x_3 ont pour domaine commun l'ensemble $\mathbb{D} = \{0,1\}$. Une valuation, c'est à dire une représentation d'une solution candidate, peut être $\{(x_1,0),(x_2,1),(x_3,1)\}$. L'ensemble $\mathcal{V}$ est isomorphe à l'ensemble des arrangements avec répétitions sur $\mathbb{D}$. La bijection φ associera donc chaque solution candidate à une permutation. La représentation est donc bien dérivée du modèle mathématique du PMU. Et sur base de la représentation, toutes les solutions candidates peuvent être générées, c'est-à-dire 2^3 triplets possibles.

4. D'autres formalisations de la représentation sont possibles Di Caro (2004); Rothlauf (2006).

2.1.3 Complexité d'un problème

La théorie de la complexité et de la calculabilité fournit le cadre théorique permettant de se prononcer sur la possibilité et la difficulté à fournir une réponse à un problème via un algorithme. Lorsque la réponse est *oui* ou *non*, le problème est un problème de décision. Si possibilité de donner une réponse il y a, la *difficulté* du problème se fonde sur l'estimation théorique du temps de calcul et de l'espace mémoire mobilisés pour obtenir algorithmiquement une réponse au problème. L'estimation est généralement exprimée en fonction de la taille de la donnée de l'instance du problème utilisée par l'algorithme. Dans le cas des POCs, la taille de la donnée dépend typiquement de la taille de l'instance du problème. Remarquons d'ailleurs que les POCs peuvent être transformés en problèmes de décision : décider s'il existe ou non une solution admissible pour une certaine limite sur sa qualité. Ainsi, un problème d'optimisation aura une complexité plus grande ou égale à celle du problème de décision associé. Et les conclusions sur la complexité des problèmes de décision peuvent être transposées aux POCs (Papadimitriou, 1994; Papadimitriou and Steiglitz, 1998).

Partant de cette notion de complexité, les problèmes peuvent être classés selon leur difficulté à être décidés algorithmiquement. On distingue ainsi un certain nombre de classes de complexité [5] en temps et/ou en espace. Parmi ces classes, les classes P et NP ont une importance toute particulière. La première, la classe P, rassemble les problèmes de décision dont une réponse peut être fournie en un temps *polynomial* par une machine *déterministe* de Turing. La seconde, la classe NP concerne les problèmes de décision auxquels une réponse peut être donnée en un temps *polynomial* sur une machine *non-déterministe* [6] de Turing. La classe NP inclue la classe P et "si $P = NP$" reste une question ouverte et centrale en informatique théorique.

Un problème NP-difficile est un problème qui est au moins aussi difficile que tous les autres problèmes NP, c'est-à-dire qu'il est possible de transformer ou réduire tous les autres problèmes NP en ce problème en un temps polynomial. A noter qu'un problème NP-difficile n'est pas nécessairement un problème NP. La sous-classe NP-complet défini un ensemble de problèmes NP qui sont NP-difficiles. Beaucoup de POCs réels sont NP-difficiles tel que le PVC alors que d'autres sont P tel que le PMU (Garey and Johnson, 1979).

5. Les problèmes classés sont nécessairement décidables.
6. Une machine non-déterministe peut-être interprétée comme effectuant parallèlement un arbre de calculs, alors qu'une machine déterministe ne peut qu'effectuer une série de calculs.

2.2 Méthode algorithmique approximative pour l'optimisation combinatoire

Les POCs sont un type de problème algorithmique, et lorsque l'on traite de leur question, vient très naturellement la question des algorithmes y répondant. Cela est d'autant plus vrai compte tenu de la complexité *NP*-difficile qu'arborent plusieurs cas pratiques de POCs. Pour beaucoup de ces problèmes, les algorithmes capables de répondre, en pratique, exactement à l'optimisation posée ne peuvent être exécutés en un temps raisonnable par rapport à l'application considérée.

Les **méthodes algorithmiques approximatives** offrent une solution pratique à ce contexte (Blum and Roli, 2003). Elles n'ont, en effet, pas pour objectif de trouver l'exact optimum quel qu'en soit le coût en temps de calcul, mais plutôt une *bonne*[7] solution en un temps raisonnable. Parmi ces méthodes, différentes classes d'algorithmes peuvent être distinguées :

- les *algorithmes d'approximation* qui garantissent l'obtention en un temps polynomial d'une approximation de l'optimum, approximation prouvée comme étant bornée par l'optimum et une limite proportionnelle à l'optimum,
- les *heuristiques* qui produisent, la plupart du temps, des bonnes solutions en appliquant des règles *de bonnes pratiques* spécifiques au problème traité, ou
- les *métaheuristiques* qui sont des stratégies génériques couplant différentes procédures, généralement des heuristiques.

Certaines de ces méthodes garantissent également une réponse dont l'optimalité sera positivement corrélée au temps de calcul octroyé, octroi connu ou non a priori. Elles répondent alors à un contexte d'optimisation dit *anytime* (Hoos and Stützle, 2004).

2.2.1 Métaheuristique

Comme définies ci-avant, les heuristiques fondent leur fonctionnement sur des règles spécifiques au problème traité. Développer de telles règles ne s'accomplit que sur la base de connaissances du problème, connaissances extraites auprès d'experts voire développées spécifiquement sur base de propriétés connues. La conception d'une heuristique nécessite un effort de développement spécifiquement à un problème donné. Un nouveau problème, un nouveau développement. Voilà qui peut être particulièrement inefficace.

7. Par bonne solution, il est entendu une réponse qui est proche de l'optimum.

Très naturellement, certains scientifiques et ingénieurs se sont attelés à trouver les moyens de factoriser ces efforts de développement. Existe-t-il des règles communes à un ensemble de problèmes ? Des règles valables dans tous les cas ? Ces questions menèrent à concevoir un nouveau type d'heuristiques non spécifiques à un problème donné et capables de fournir une réponse à plusieurs problèmes : les **métaheuristiques**. Elles sont, aujourd'hui, à compter parmi l'état de l'art des méthodes algorithmiques approximatives dédiées aux POCs (Blum and Roli, 2003; Hoos and Stützle, 2004). Comme exemple, nous pouvons, entre autres, citer les Algorithmes Génétiques (AGs) (Holland, 1975), les algorithmes de colonies de fourmis (Dorigo and Stützle, 2004), le recuit simulé (Kirkpatrick et al., 1983), ou encore l'optimisation par essaims particulaires (Eberhart and Kennedy, 1995).

Nonobstant les différents formalismes tentant de définir précisément la famille d'algorithmes que sont les métaheuristiques, Blum and Roli (2003) les classent et les décrivent selon les critères de *diversification* et d'*intensification*. La diversification désigne un processus de récolte de nouvelles informations en explorant de nouvelles aires [8] de l'espace de recherche. L'intensification quant à elle désigne un processus mettant à profit l'information récoltée pour circonscrire la recherche dans les aires intéressantes. Partant de cette définition descriptive, les **métaheuristiques** sont des stratégies de haut niveau combinant des procédures, généralement des heuristiques, pour répondre à un problème d'optimisation. Ces stratégies guident la recherche en répartissant les coûts computationnels incombés entre les procédures favorisant la diversification ou l'intensification.

Les procédures combinées au sein d'une métaheuristique sont appelées *opérateurs*. Suivant la philosophie originale qui a enfanté les métaheuristiques, ces opérateurs ne sont pas conçus et développés spécifiquement à un POC. Mais, ils sont souvent plus attachés à une famille de POC et/ou un type de représentation. Cette approche respecte le scénario de la boîte noire. Dans ce scénario, l'instance du POC est considérée comme une boîte noire que l'on ne peut interroger que de la manière suivante : "Quelle est la valeur objective $f(s)$ de la solution candidate $s \in S$?" En pratique, l'obtention de l'état de l'art des résultats nécessite bel et bien la spécialisation des opérateurs.

Les métaheuristiques sont donc des stratégies de mise en oeuvre d'un ensemble donné d'opérateurs. Parmi les différentes stratégies, deux types peuvent être distingués : le parcours et la population. Dans le premier cas, la stratégie est fondée sur le parcours de l'espace de recherche au moyen de l'évolution itérative d'une seule solution candidate. Dans le second cas, il est manipulé parallèlement une population, autrement dit un ensemble

8. Une aire est un sous-ensemble de l'espace de recherche. Elle rassemble des solutions candidates qui sont proches.

de solutions candidates, à chaque itération. Mais les stratégies ne s'arrêtent pas à ces seules considérations. Chaque mécanisme s'avère être une combinaison balancée, subtile et unique de diversification et d'intensification. Et selon le POC considéré, la taille de l'instance, ou encore les propriétés de son espace de recherche, cette combinaison sera plus ou moins adaptée (Blum and Roli, 2003).

Le principe sous-jacent des métaheuristiques est, comme précédemment dit, de généraliser suffisamment l'algorithme d'optimisation que pour le détacher d'un POC donné. Très logiquement, on peut se poser la question d'une métaheuristique capable de résoudre tous les POCs. Cette question est à la base du théorème du "No Free Lunch" (Wolpert and Macready, 1997). Ainsi, peut-on affirmer qu'un tel algorithme n'existe pas pour l'ensemble de tous les POCs possibles ou que deux algorithmes seront en moyenne aussi performant l'un que l'autre sur l'ensemble de tous les problèmes possibles. Bien qu'intéressant, ce théorème n'est pas réellement significatif pour les cas pratiques réels qui ne sont qu'un sous-ensemble de tous les problèmes possibles. En effet, le théorème du "No Free Lunch" n'est plus nécessairement vrai lorsque l'on considère un sous-ensemble (Igel and Toussaint, 2003). Par conséquent, lorsque l'on s'attache à traiter une famille donnée de POC, la conception d'un algorithme adéquat et idéal reste a priori tout à fait possible.

2.2.2 Représentation et métaheuristique

La section 2.1.2.3 a exposé comment il était possible de représenter un POC de manière constructive. Aussi, la **représentation** va-t-elle déterminer comment les solutions candidates sont encodées et comment les opérateurs d'une métaheuristique peuvent manipuler leurs composantes, autrement dit les variables de représentation (Milano and Roli, 2004). La représentation constitue donc le modèle de travail qu'utilise une métaheuristique. Elle influe donc conséquemment l'implémentation d'une métaheuristique, ne fut-ce que par l'information sur problème qu'elle définit implicitement. Même dans le scénario de boîte noire, la représentation constitue, en effet, une connaissance a priori fournie à la métaheuristique. La représentation est en quelque sorte le filtre au travers duquel la métaheuristique perçoit le POC et agit sur lui.

Ainsi, l'efficacité d'une métaheuristique dépendra entre autres de la représentation utilisée. Cet état de fait est bien connu dans la communauté des AGs (Rothlauf, 2006). En pratique, l'obtention de bons résultats peut notamment passer par l'exploitation de spécificité d'un problème. C'est, par exemple, le cas de l'opérateur de Croisement par Recombinaison d'Arêtes (CRA, cf l'Appendice C) qui exploite l'adjacence, une propriété importante

du PVC (Moraglio and Poli, 2005). La représentation est donc une autre manière de spécialiser une métaheuristique à un POC donné : elle la *dé*-généralise.

2.2.3 Graphe de voisinage

Le fonctionnement d'une métaheuristique repose notamment sur la représentation choisie et les opérateurs utilisés. Chaque opérateur définit alors une métrique pour la représentation (Moraglio, 2007). Un opérateur et une représentation définissent une *relation de voisinage* N sur l'espace de recherche. Cette relation binaire n'est pas nécessairement symétrique[9] sur S, mais nous nous limiterons à ce cas. Cette relation N est définie par une partie $\mathcal{N}$ de $S \times S$. Deux solutions candidates s, $s' \in S$ seront alors dites voisines sNs' pour autant que le couple (s,s') appartient à $\mathcal{N}$. La relation de voisinage N peut être associée canoniquement à une fonction $n : S \to 2^S$ tel que $n(s)$ nous donne le voisinage de s, l'ensemble $\{s' \in S | sNs'\} \subseteq S$.

Cette relation N induit sur l'espace de recherche S un graphe simple (S,N) : le **graphe de voisinage** ν (Hoos and Stützle, 2004). Les sommets de ν ne sont autres que les solutions candidates, et les arêtes les couples $(s,s') \in \mathcal{N}$. Un degré de voisinage d peut être défini pour le graphe ν comme étant la quantité $|\mathcal{N}|$.

Ce graphe de voisinage décrit quels sont les mouvements possibles de l'opérateur considéré dans l'espace de recherche muni d'une représentation donnée. Comme Hoos and Stützle (2004) l'affirment, le processus d'une métaheuristique peut en effet être vu comme une marche sur le graphe de voisinage associé à chacun de ses opérateurs. Le processus ne peut sortir de cette structure.

2.2.4 Algorithme évolutionniste

Après avoir introduit plus généralement les métaheuristiques dans la Sec. 2.2.1, la présente section s'attachera à décrire plus précisément la famille de métaheuristiques utilisées dans cette thèse : les **Algorithmes Evolutionnistes** (AEs). Les AEs désignent une famille de métaheuristiques s'inspirant de la théorie de l'évolution et furent introduits pour la première fois par Holland (1975) sous la forme des AGs. La grande majorité des AEs manipulent des population de solutions candidates par le biais de mécanismes dérivés des principes de l'évolution naturelle introduite par Darwin (1872) :

– les caractéristiques d'un individu dépendent en grande partie de son code génétique,

9. Par conséquent, xNy est équivalent à yNx.

Algorithm 1 Algorithme Evolutionniste (AE).

1: **construction** d'une **population initiale**
2: **évaluation** de la population initiale
3: **while** critère d'arrêt non satisfait **do**
4: **sélection** d'une partie de la population
5: **reproduction** des individus sélectionnés
6: **mutation** de la descendance
7: **évaluation** de la descendance
8: **génération** de la nouvelle population
9: **end while**

- les individus cohabitent dans un environnement en nombre limité au sein d'une population,
- au sein de cette population, les individus se reproduisent et transmettent une partie de leurs caractéristiques à leur descendance, et
- les individus les plus adaptés se reproduisent plus souvent.

Partant de ces principes, les AEs peuvent être décrits comme suit (voir également l'Alg. 1) (Bäck, 1996; Bäck et al., 1997; Goldberg, 1989; Holland, 1975; Mitchell, 1996; Rothlauf, 2006). Tous les AEs font évoluer une population de solutions candidates, appelés *individus* ou *phénotypes*. Le POC posé constitue l'environnement de la population et chaque individu peut ainsi être évalué sur son adaptation à cet environnement, adaptation déterminant ses chances de survie et/ou de reproduction. Pour ce faire, la qualité de l'individu est calculée à l'aide d'une *fonction d'adaptation*. Dans les cas les plus évidents, cette fonction d'adaptation est simplement la fonction objective du POC concerné. Au tout début d'un AE, les individus sont créés aléatoirement ou non et ensuite évalués pour former une population *initiale* (voir les lignes 1 et 2 de l'Alg. 1). L'évolution peut ensuite démarrer à proprement dite pour former de nouvelles *générations* d'individu. A chaque génération, certains individus sont sélectionnés sur base de leur qualité d'adaptation. De par leur reproduction, une descendance voit alors le jour (voir les lignes 4 et 5 de l'Alg. 1). Ces nouveaux individus subissent éventuellement une mutation et leur qualité d'adaptation est évaluée (voir les lignes 6 et 7 de l'Alg. 1). L'ancienne population est alors remplacée totalement ou en partie par la descendance produite (voir la ligne 8 de l'Alg. 1). Et le cycle recommence jusqu'à que certains conditions soient rencontrées comme un nombre maximum de générations ou un niveau donné d'adaptation (voir la ligne 3 de l'Alg. 1).

Communément, les AEs respectent le scénario de la boîte noire et ne font donc aucune hypothèse sur le POC traité. Ils constituent effectivement un cadre très généraliste, et malgré cela, ils peuvent générer de très bonnes solutions candidates. Cette réussite requiert une attention toute particulière

sur deux points clés : la fonction d'adaptation et la représentation génétique des individus.

2.2.4.1 Fonction d'adaptation

La fonction d'**adaptation** ou d'**évaluation** $\hat{f} : S \rightarrow \mathbb{R}$ permet d'évaluer la qualité de l'individu. En terme d'optimisation, il s'agit bien entendu d'évaluer la qualité d'une solution candidate. Idéalement, un optimum s^* pour $\hat{f}$ restera un optimum pour la fonction objective f de l'instance du problème considéré. Si la fonction objective f est connue pour l'instance considérée du POC, la fonction d'adaptation n'est autre que la fonction objective. L'individu le plus adapté correspond alors trivialement à l'optimum de la fonction f. Par contre, si la fonction objective n'est pas explicitement connue, la fonction d'adaptation sera définie de telle manière à quantifier la qualité de l'individu à répondre à une instance donnée de POC.

2.2.4.2 Représentation génétique

En biologie, il existe une différence entre le code génétique d'un individu et son expression, qui n'est autre que l'individu lui même. Le code génétique est contenu dans les chromosomes. Un chromosome peut être décomposé en un ensemble de régions, les gènes, qui ont un rôle d'expression précise. Pour chaque gène, il existe différentes versions possibles, les allèles, et donc différentes expressions. Le *génotype* est la composition allèlique de tous les gènes d'un individu. Le *phénotype* désigne, lui, l'expression exacte et observable que constitue l'individu, son apparence, ses caractéristiques. Il dépend du génotype de l'individu.

Cette distinction est conservée dans l'analogie algorithmique qu'est un AE. Le génotype n'est autre que la représentation des solutions candidates sur laquelle agiront les différents opérateurs. Le phénotype, quant à lui, permet d'exprimer ou de développer la solution candidate afin de l'évaluer. La représentation génétique est une représentation indirecte qui nécessite une transformation de la représentation des individus et leur interprétation en terme d'adaptation.

Rothlauf (2006) fournit une formalisation précise pour définir la représentation génétique. L'espace de recherche génotypique Φ_g est l'ensemble des représentations possibles des différents individus. Les opérateurs d'un AE agissent dans cet ensemble Φ_g. Le chromosome est défini comme un n-uplet d'allèles $x_g \in \Phi_g$ de longueur n. L'espace de recherche phénotypique Φ_p consiste en l'ensemble isomorphe à l'espace de recherche des individus exprimés, autrement dit l'espace de recherche S. L'espace Φ_p correspond typiquement au cas de la représentation des POCs comme définie dans la

Sec. 2.1.2.3, c'est-à-dire l'ensemble $\mathcal{V}$. Une application $\hat{f_g}$ est définie de Φ_g vers Φ_g afin d'exprimer les chromosomes qui représentent les différentes individus. Selon l'application $\hat{f_g}$ définie, il est possible que différents chromosomes désignent le même individu. La fonction d'adaptation est définie sur l'espace phénotypique $\hat{f_p} : \Phi_p \rightarrow \mathbb{R}$. Et l'évaluation d'un chromosome consiste en la composition de ces deux fonctions : $\hat{f} = \hat{f_p} \circ \hat{f_g}$. Si l'application $\hat{f}$ est une bijection, la représentation est dite directe et les opérateurs agissent directement dans l'espace phénotypique. Nous retrouvons le cas précédemment décrit à la Sec. 2.1.2.3.

2.2.4.3 Opérateurs évolutionnistes

La manipulation des individus s'effectue au moyen des opérateurs de sélection et de variations. Les opérateurs de sélections concernent la sélection des individus destinés à se reproduire et le remplacement des anciens individus par la nouvelle descendance. Ils agissent donc dans l'espace phénotypique. Les opérateurs de variation permettent la reproduction et la mutation des individus. Ils agissent dans l'espace génotypique.

Sélection et remplacement Pendant la phase de sélection, soit une partie de la population est remplacée par la nouvelle descendance soit la totalité. La nouvelle descendance est produite sur base du clonage et/ou de la reproduction des individus sélectionnés. La sélection s'opère généralement en fonction de la qualité d'adaptation des individus. Mais d'autres modes sont possibles, en particulier ceux qui vont assurer une certaine diversité dans la population, évitant ainsi une convergence prématurée autour d'un seul individu mieux adapté.

Mutation Une mutation est un opérateur insérant de petites variations dans le génotype d'un individu. C'est un opérateur important pour la recherche locale ou pour réhabiliter des allèles détruits par la reproduction. Une mutation qui changerait beaucoup trop d'allèles dégraderait l'évolution en une recherche aléatoire.

Recombinaison La recombinaison ou croisement est l'opérateur assurant l'enjambement des génotypes des parents lors d'une reproduction. Il assure donc l'héritage de certains gènes entre les parents et sa descendance. La recombinaison consiste donc principalement en la recopie exclusive des parties chromosomiques provenant des parents selon une stratégie définie par l'opérateur lui-même.

2.2.5 Paysage et structure

Il est une métaphore particulièrement intéressante, en théorie de l'évolution, qui permet de visualiser les dynamiques évolutionnistes de spéciations [10] : le **paysage d'adaptation**. Cette métaphore, introduite par Wright (1932) dans les années 1930, a également été récupérée par la communauté des AEs. Dans ce cadre, le paysage d'adaptation est une surface qui s'étend par dessus l'espace génotypique (Watson, 2006). A chaque génotype est ainsi associé un niveau d'adaptation constituant un point de la dite surface. Le voisinage des points de la surface correspond au voisinage des génotypes correspondants induit par des variations génétiques. Si l'on suit le formalisme introduit précédemment provenant majoritairement de Hoos and Stützle (2004), le paysage d'adaptation peut alors être défini formellement comme étant la paire $(v, \hat{f})$. Il est donc clair que sans métrique de proximité, le paysage perd toute *structure* et donc toute signification visuelle.

En optimisation combinatoire, de nombreux auteurs (Blum and Roli, 2003; Borenstein and Poli, 2006; Glover and Kochenberger, 2003; Hoos and Stützle, 2004; Jones, 1995; Reeves, 1999; Reidys and Stadler, 2002) ne font pas la distinction entre les concepts de paysage d'adaptation et de structure. La structure ou topologie du paysage ne dépend pas du niveau d'adaptation d'un individu particulier mais bien de la relation qui existe entre les différents individus en terme d'adaptation (Borenstein and Poli, 2006). En somme, le paysage d'adaptation traduit visuellement la structure d'un POC muni d'une certaine représentation sous l'action un opérateur donné. Mais d'autres types de structure existent, autrement dit d'autres propriétés de relation entre les solutions candidates d'une instance de POC, comme les symétries, les modalités multiples ou les colonnes vertébrales (*backbones* en anglais) (Hoos and Stützle, 2004; Kallel et al., 2001; Reidys and Stadler, 2002; van Hemert, 2006). Et ces structures n'apparaîtront pas toujours sur le paysage d'adaptation, selon les opérateurs et représentations considérés.

2.3 Conclusion

Une instance de POC est définie comme une paire (S, f), où l'espace de recherche S est discret et fini, dont on cherche l'extremum global de la fonction f. La représentation d'une telle instance se définit comme la bijection $\varphi : S \to \mathcal{V}$ faisant correspondre à toute solution candidate $s \in S$ un n-uplet de valeur $c(x) \in \mathbb{D}_x$ pour chaque variable $x \in X$. Une instance représentée peut donc se définir comme le triplet $(X, \mathcal{V}, f)$.

10. La spéciation désigne le processus évolutif par lequel de nouvelles espèces vivantes apparaissent.

Les métaheuristiques, un type général et malléable de méthodes algorithmiques approximatives, fonctionnent selon des stratégies de haut niveau combinant des procédures unitaires appelées les opérateurs. Le principe sous-jacent des métaheuristiques est de se généraliser suffisamment que pour être détaché d'un POC donné. Les opérateurs agissent sur les solutions candidates par l'intermédiaire d'une représentation, et de ce fait la représentation peut être vue comme le modèle de travail qu'utilise une métaheuristique. Ainsi, le processus de recherche d'une métaheuristique peut être modélisé par le graphe de voisinage $\nu = (S, N)$. Ce graphe dépend de la relation de voisinage qui est le produit de la conjonction d'un opérateur et d'une représentation. Il décrit de ce fait tous les mouvements possibles de l'opérateur impliquant N.

Le paysage d'adaptation [11] est défini comme la paire (ν, f). Il traduit visuellement la structure d'une instance de POC muni d'un certain représentation sous l'action d'un opérateur donné. Mais ce n'est pas la seule structure qui peut être décrite. Lato sensu, la structure est un terme qui englobe toute propriété de relation entre les solutions candidates d'une instance de POC.

11. Nous choisissons ici d'user de la terminologie propre à toute l'optimisation combinatoire pour ne pas surcharger inutilement le texte.

3
Une Gestalt en optimisation combinatoire

Après avoir établi le contexte de la dissertation, nous allons au sein de ce chapitre d'une part introduire la source de notre inspiration, la psychologie de la Gestalt, et d'autre part discuter son concept analogue appliqué aux métaheuristiques.

Ainsi dans une première partie, nous commençons par introduire la psychologie de la Gestalt. Nous nous attardons en particulier sur la systémique sur laquelle elle base sa théorie. Ce qui nous amène à toucher un mot à propos des lois de l'organisation qu'elle établit pour expliquer la mise en forme de la perception. Et nous terminons sur son extension aux processus cognitifs de résolution de problème. C'est sur cette base que nous construisons dans la suite l'analogie de la Gestalt dans le cadre formé par l'optimisation combinatoire et les métaheuristiques.

Dans une deuxième partie, nous construisons à proprement dit cette analogie. Un parallèle est alors établi entre l'humain résolvant un problème et une métaheuristique traitant une certaine instance de POC. Fort du formalisme exposé au Chapitre 2, nous identifions la représentation comme le moyen perceptif de la métaheuristique et expliquons pourquoi l'altération de la représentation suffit à modifier le processus de recherche. Ceci étant, nous développons l'analogie en définissant une Gestalt canonique en termes de l'agrégation de variables et en expliquant l'impact de cette reformulation de représentation sur les actions des opérateurs utilisés. Ceux-ci se voit en effet contraint de respecter l'intégrité de la Gestalt créée. Ensuite, nous décrivons les principes d'un mécanisme chargé de la construction d'une

Gestalt de représentation en nous inspirant des principes dégagés par la psychologie de la Gestalt appliquée aux processus cognitifs de résolution de problème.

Nous terminons le chapitre par une troisième et dernière partie discutant de l'approche générale dans laquelle s'inscrit notre analogie de la Gestalt en optimisation combinatoire : une méta-modélisation dynamique et en ligne pour l'assistance d'une métaheuristique dans son processus de recherche. En quoi une Gestalt de représentation est-elle une méta-modèle de recherche ? Que signifie dynamique et en ligne ? Telles sont les questions auxquelles nous répondons dans cette dernière partie.

In fine, le chapitre se conclut en reprenant les éléments importants qui servirons de pierre d'achoppement au chapitre suivant. Abordons maintenant la source principale de notre inspiration, la psychologie de la Gestalt.

3.1 La psychologie de la Gestalt

La **psychologie de la Gestalt** ou psychologie de la forme fournit une théorie générale tentant de comprendre certains processus mentaux telle que la vision ou la résolution de problèmes comme le produit d'une systémique. Selon Max Wertheimer, Wolfgang Kölher ou Kurt Koffka (Ash, 1995), la perception ne se résume ainsi pas en un simple ensemble de stimuli individuels. Les différents stimuli se combinent, s'arrangent et se divisent afin de se grouper en des agrégats signifiants. L'environnement est alors perçu en terme de sa structure, par le biais de propriétés macroscopiques (Wertheimer, 1938). Une totalité ne peut se réduire à la simple somme des stimuli. La Figure 3.1 illustre cette importante propriété : le chien n'est visible que lorsque chaque tache prend une signification par rapport aux autres. De même, la société humaine est plus qu'un ensemble d'être humains, ou un être humain plus qu'une agglomération de cellules, ou encore une molécule d'eau est plus qu'un atome d'oxygène et deux atomes d'hydrogènes. Cet holisme peut se résumer par la maxime "le tout est plus que la somme des parties". De même, le tout, le contexte, influence le signifiant d'une partie. Un objet ne prend tout son sens que par rapport aux objets environnants. Autrement dit, une partie est plus que son individualité (Ash, 1995).

Ainsi, n'existe-t-il pas de perception isolée dans la Gestalt. Il est en effet admis que certaines formes et structures sont favorisées sur d'autres dans le processus de vision. Nombreuses sont les théories tentant de répondre à ce fait. La psychologie de la Gestalt fut l'une des théories les plus influentes au milieu du vingtième siècle et elle inspire encore aujourd'hui de nombreuses recherches contemporaines Scholl (2001).

FIGURE 3.1 – L'*Image du Chien* est une image célèbre par R.C. James (Gregory, 1970). Elle permet d'illustrer la maxime "le tout est plus que la somme des parties" appliquée à la perception visuelle. Le chien, un dalamatien, renifle le sol au centre de l'image, son corps se trouvant sur la partie droite de l'image.

3.1.1 Les lois de l'organisation de la perception

Afin d'expliquer la structuration de la perception et en particulier les agrégations, les gestaltistes ont postulé des lois de l'organisation (voir Fig. 3.2) : proximité, similarité, fermeture, simplicité, . . . Koffka (1935) a suggéré que ces lois étaient coordonnées par la loi de la simplicité ou de *Prägnanz* : parmi plusieurs organisations géométriques possibles, prédomine celle possédant la meilleure, la plus simple et la plus stable forme.

A juste titre, certains, comme Bruce et al. (1996), font remarquer qu'une telle formulation n'est que trop vague et imprécise que modéliser les processus de perception. La psychologie de la Gestalt ne fournit en effet qu'une théorie descriptive et conceptuelle, loin d'un modèle formel précis. Néanmoins, cela n'a nullement empêché certains d'en déduire des modèles fonctionnels de vision artificielle (Dastani, 1998; Zhu, 1999).

3.1.2 La Gestalt appliquée à la résolution de problème

Comme il a été mentionné plus tôt, la Gestalt s'applique également à expliquer le processus cognitif de résolution de problème, une étude principalement initiée par Wertheimer (1959). Selon ce dernier, le succès d'un tel processus cognitif réside dans la capacité à embrasser la structure

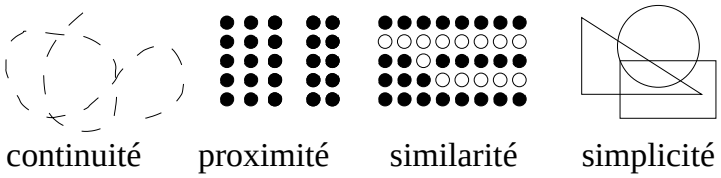


FIGURE 3.2 – Les lois de l'organisation sont des principes génériques définis par la Gestalt pour comprendre l'obtention de certaines agrégations dans le cadre de la résolution de problème ou la perception visuelle. Cette figure représente quatre de ces lois : les lois de continuité, de proximité, de similarité et de simplicité. Selon la loi de continuité, les agrégats sont vus comme un tout et les discontinuités tendent à être ignorées. Selon celle de proximité, les objets proches temporellement et/ou spatialement peuvent être classés ensemble. Selon celle de similarité, les objets similaires tendent à être associés et regroupés ensemble. La loi de la simplicité ou de Prägnanz coordonne les lois de l'organisation en indiquant que la configuration la plus simple et la plus stable est préférée.

totale du problème. Par exemple, un enfant peut comprendre le calcul de surface de quadrilatère ou de triangle en se basant sur celui du rectangle (voir Fig. 3.3), et replaçant ainsi dans son contexte les principes de hauteur et de base. On retrouve ici, comme dans le cas de la perception, l'approche systémique propre à la Gestalt.

Wertheimer (1959) propose ainsi trois principes fondamentaux à la résolution de problème et son apprentissage, principes que nous dénommerons de la manière suivante :

1. l'investigation,
2. la perturbation, et
3. la contextualisation.

L'investigation consiste en l'apprentissage et la reconnaissance de la structure d'un problème, c'est-à-dire la relation qui existe entre les différents éléments constitutifs. L'identification de cette relation permet de construire une Gestalt, c'est-à-dire une signification complète par rapport au problème global. Cette investigation est un processus dynamique qui doit pouvoir adapter cette Gestalt, en particulier lors de l'introduction d'un nouvel élément ou de tout autre type de perturbation. Elle utilise les lois de l'organisation pour construire ou modifier une Gestalt.

La perturbation, le deuxième principe, constitue alors un stimulus enclenchant des adaptations et/ou des innovations lors de l'investigation. Ce stimulus consiste principalement en l'introduction d'erreurs dans la structure actuellement acquise avec pour objectif de tester sa stabilité tout en assurant une certaine flexibilité.

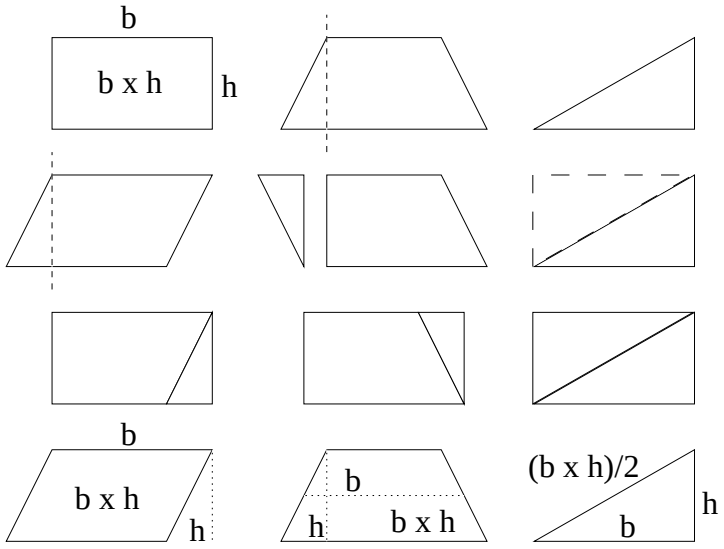


FIGURE 3.3 – Comment un enfant résout-il l'air d'un parallélogramme, d'un trapèze ou d'un triangle en ne connaissant que la règle pour calculer l'air d'un rectangle ? Selon Wertheimer (1959), l'enfant est capable d'identifier les isomorphismes entres les différents polygones. En transformant par exemple le parallélogramme en un rectangle, l'enfant peut ainsi calculer son air et contextualiser les principes de hauteur et de base.

Le dernier principe n'est autre que l'exploitation d'une structure du problème dans sa résolution : la contextualisation. Elle se réalise en formant la Gestalt, autrement dit en organisant et agrégeant, ensemble, les éléments constitutifs. Elle nourrit d'une part l'investigation, mais également la résolution, en tant que telle, du problème. A noter que lors de cette phase, certaines lois de l'organisation peuvent agir, comme par exemple la loi de Prägnanz.

3.2 Une Gestalt pour les métaheuristiques

La psychologie de la Gestalt suggère une théorie pour expliquer chez l'humain certains processus complexes, en particulier la résolution de problème : le problème se résout par la construction naturelle d'une Gestalt, c'est-à-dire comprendre, saisir ensemble, tous les composants du problème dans le contexte du dit problème. Reprenant le contexte de la présente thèse, les métaheuristiques et l'optimisation combinatoire, une analogie peut être établie entre elles et les humains : les métaheuristiques sont des techniques qui cherchent à résoudre, du moins approximativement, des instances de problème d'optimisation (voir la Sec. 2.2.1), tout comme les humains mettent inconsciemment en oeuvre tous les jours des techniques pour résoudre de telles instances. Compte tenu de cette analogie, peut-on s'inspirer des

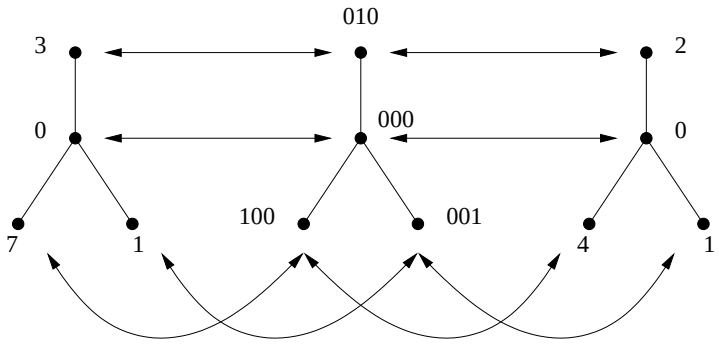


FIGURE 3.4 – La figure illustre l'impact de la représentation choisie sur la proximité des solutions candidates. En particulier, il est considéré ici un espace de recherche contenant les nombres entiers de 0 à 7, une représentation sur trois variables binaires, deux modes différents de valuation que sont l'encodage binaire classique et de Gray des entiers, et un opérateur de basculement d'un bit. Le graphe du milieu représente le voisinage de la solution candidate **0** représentée par **000** quelque soit le mode de valuation. Quant au graphe de gauche, il représente le voisinage de **0** induit par l'encodage binaire réfléchi. Le graphe de droite représente le voisinage de **0** induit par l'encodage binaire classique. Dans le premier cas, les voisins de **0** sont **1**, **3** et **7**. Alors que dans le second cas, ils sont **1**, **2** et **4**.

concepts introduits par la psychologie de la Gestalt pour aider la métaheuristique dans sa tâche première, la recherche d'une bonne solution ? Cette question constitue le point de départ de la présente thèse.

3.2.1 Représentation comme perception du problème

Avant de pouvoir déterminer ce qu'est une Gestalt dans le contexte de l'optimisation combinatoire, il est nécessaire de déterminer comment une métaheuristique perçoit une instance de POC. Comme vu précédemment dans la Sec. 2.2.2, une représentation φ d'une instance de POC constitue le modèle de travail qu'utilise la métaheuristique. Et en ce sens, une représentation détermine une certaine perception du problème, c'est-à-dire une certaine interprétation.

En effet, la représentation choisie détermine en partie le graphe de voisinage (voir 2.2.3), un graphe qui décrit les mouvements possibles d'un opérateur dans l'espace de recherche. Il suffira de changer la représentation, comme par exemple les valuations, pour que le graphe de voisinage change, et par conséquent pour que la manière de chercher change. La Figure 3.4 illustre cet impact : selon que les solutions candidates, des nombres entiers, soient représentées au moyen d'un codage binaire classique ou de Gray, le basculement d'un bit induit des voisinages différents (Rowe et al., 2004).

3.2.2 Une Gestalt canonique : l'agrégation de variables

La métaheuristique perçoit et manipule donc des solutions candidates à une instance de POC au travers d'une certaine représentation. Mais qu'est-ce qu'un *tout* ? Que sont les *parties* ?

La Section 2.1.2 nous a appris qu'une instance de POC était définie comme étant la paire (S, f). Cette paire est notre *tout*. Pour les *parties*, par contre, plusieurs possibilités peuvent être envisagées : les solutions candidates, ou encore les variables. De par leur nombre, les solutions candidates sont à rejeter comme possibilité. En effet, une Gestalt ne pourrait être formée qu'en les considérant toutes. Malgré leur nombre restreint, les variables permettent de générer toutes les solutions candidates par le biais de valuations. Elles sont, par définition de la représentation, *parties* constitutives et signifiantes de la représentation d'une instance et attachées à toute l'instance. Elles constituent donc de simples *parties* de choix.

Une Gestalt, comme expliqué au début de ce chapitre, est à la fois une non individualisation, un rassemblement signifiant et une mise en relation des parties. Dans cette optique, la métaheuristique doit alors non plus manipuler des variables individuellement, mais selon une certaine mise en forme et regroupement. Les opérateurs de la métaheuristique travaillent alors sur des agrégats insécables de variables, formant des variables macroscopiques. Une Gestalt canonique [1] de la représentation d'une instance de POC revient à l'agrégation de variables en de nouvelles entités constituantes pouvant être manipulées par les opérateurs en tant que telles.

3.2.3 Une Gestalt comme abstraction de la recherche

La Gestalt canonique d'une représentation consiste à agréger les variables en des groupuscules insécables, définissant par là même une nouvelle représentation. Les opérateurs de la métaheuristique ne peuvent dès lors plus travailler exactement de la même manière. Là où ils manipulaient des variables individuellement, ils doivent désormais en manipuler des ensembles indivisibles. Il est alors nécessaire de définir une opération pour cet ensemble. Cette opération d'ensemble s'obtient par l'application de l'opération fondamentale par variable à toutes variables constituant l'agrégat. L'opérateur est donc naturellement généralisé par la délégation de son opération aux parties de l'ensemble qu'il manipule. Et dès lors, l'application d'une Gestalt change le graphe de voisinage.

Illustrons cette conséquence en prenant, par exemple, une instance de problème représentée initialement par trois variables binaires (x_1, x_2, x_3) et le basculement d'un bit comme opérateur. Tel que nous pouvons le voir à

1. Par canonique, il est entendu le plus simple et porteur de sens.

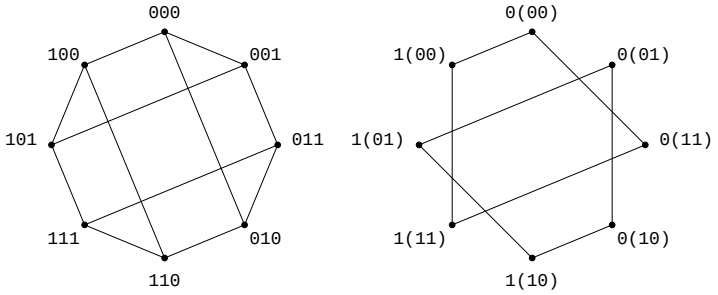


FIGURE 3.5 – Il est illustré sur cette figure l'impact d'une Gestalt sur le graphe de voisinage. Pour ce faire, nous considérons une instance de problème représentée par trois variables binaires et le basculement d'un bit comme opérateur. Sur la gauche, le graphe de voisinage initial est représenté. Et sur la droite, sa transformation suite à l'application de la Gestalt agrégeant les variables x_2 et x_3.

la Fig. 3.5, si le point de départ considéré est $(0,0,0)$, ses trois seuls voisins possibles sous cet opérateur sont $(0,0,1)$, $(0,1,0)$ et $(1,0,0)$. Définissons maintenant une Gestalt telle que les variables x_2 et x_3 soient considérées comme un nouvel ensemble à part entière, soit le couple (x_2,x_3). L'application du basculement d'un bit à cet agrégat implique son application à chacune de ses parties, les variables x_2 et x_3. Dès lors, les voisins de $(0,0,0)$ ne sont plus qu'au nombre de deux (voir la Fig. 3.5) : $(0,(1,1))$ et $(1,(0,0))$. A noter que les solutions candidates $(0,0,1)$ et $(0,1,0)$ ne sont plus à compter parmi le voisinage et que le voisin $(0,(1,1))$ est un nouveau venu. Ce changement de voisinage concerne bien entendu toutes les solutions candidates comme on peut le voir sur Fig. 3.5. Et parfois à tel point qu'il n'existe plus aucun chemin entre certaines solutions candidates, comme entre $(0,0,1)$ et $(0,0,0)$. Le graphe de voisinage est bel et bien altéré par une Gestalt de la représentation.

Une Gestalt de la représentation implique une modification plus ou moins grande du graphe du voisinage de par la généralisation de l'opérateur aux agrégats. Pour conséquence, certains chemins sont supprimés, d'autres sont synthétisés. Dans tous les cas, la transformation du graphe du voisinage diminue son degré de voisinage puisque, par l'agrégation, les possibilités de l'opérateur sont réduites. En ce sens, on est face à une simplification synthétique et globale du graphe de voisinage, et de fait une simplification de la recherche dirigée par un opérateur. Ainsi, non seulement une Gestalt de la représentation est une abstraction[2] structurelle de par l'agrégation de variables, mais elle est également une abstraction fonctionnelle du processus de recherche.

2. Etant entendu par abstraction une simplification guidée par un trait ou un autre.

3.2.4 Construction d'une Gestalt

Dans les sections précédentes, nous avons montré comment transposer le concept de la Gestalt dans le présent contexte, qui, rappelons-le, traite des métaheuristiques, et en particulier les AEs, appliquées à l'optimisation combinatoire : une Gestalt canonique consiste en une agrégation des variables en des ensembles, des groupes insécables et atomiques. Les opérateurs manipulent alors ces agrégats comme de nouvelles variables à part entière. Implicitement, une Gestalt de la représentation cause également une abstraction de la recherche. Mais comment cette Gestalt peut-elle être construite ?

C'est ici que les principes de la Gestalt appliqués à la résolution de problème, introduits à la Sec. 3.1.2, interviennent. Et par analogie, il s'agit donc de construire un algorithme assurant l'investigation, la perturbation et la contextualisation pour construire la Gestalt.

3.2.4.1 Investigation

Lors de l'investigation, l'objectif visé est principalement la compréhension du problème en terme de sa structure. Dans notre contexte, la structure est un terme, nous l'avons vu dans la Sec. 2.2.5, englobant tout type de propriété générale du problème. Lato sensu, une Gestalt de représentation est bien l'expression d'une structure particulière. Il conviendra donc de construire différentes compréhensions du problème, différentes Gestalts, et de trouver celle qui permettra la métaheuristique de "mieux" chercher. Cette investigation se nourrit des avancées de la métaheuristique et s'effectue donc en parallèle de la métaheuristique : la compréhension du problème s'effectue en même temps que sa résolution.

Par analogie, la construction des différentes Gestalts se baserait sur des lois d'organisation. Compte tenu de l'ignorance du problème, il est difficile de définir a priori des propriétés telle que la proximité ou la similarité, voire d'établir d'autres propriétés pertinentes pouvant servir la construction d'une Gestalt. Pour résoudre ce problème, la construction se basera sur des règles génériques dirigées par la contextualisation. Cette dernière fournira une mesure qui confirmera ou non la bonne qualité d'une Gestalt.

3.2.4.2 Perturbation

Pour pouvoir trouver de nouvelles Gestalts, l'investigation se reposera, entre autres, sur la perturbation. La perturbation introduit explicitement des petites variations ou erreurs dans la Gestalt afin d'assurer une dynamique du type essai-erreur permettant l'innovation et l'adaptation. Ce principe im-

portant permet d'éviter de former une Gestalt valable seulement pour une partie du problème, c'est-à-dire un sous-ensemble de solutions candidates déjà visitées.

3.2.4.3 Contextualisation

Le dernier principe, la contextualisation, n'est autre que la mise en pratique d'une Gestalt, et son évaluation. La mise en pratique d'une Gestalt revient à sa simple utilisation par la métaheuristique. Les opérateurs doivent alors en tenir compte dans leur procédé et respecter les agrégats définis. L'évaluation, quant à elle, permet de mesurer la pertinence de la Gestalt à bel et bien aider la métaheuristique dans sa recherche. Cette mesure est ensuite utilisée par l'investigation, confirmant ou invalidant la pertinence de la Gestalt.

Cette évaluation permettant de renforcer ou non l'adéquation d'une Gestalt peut être considérée comme une loi de Prägnanz. Il s'agira en effet de sélectionner les Gestalts selon une mesure de qualité. Ceci revient à sélectionner implicitement une certaine organisation reflétant les propriétés pertinentes pour le problème.

3.3 Une méta-modélisation dynamique et en ligne

La Section 3.2 précédente a montré comment transposer les concepts inhérents à la psychologie de la Gestalt à l'optimisation combinatoire menée par une métaheuristique. Dans l'analogie qui y a été construite, une Gestalt de représentation consiste à définir une nouvelle représentation isomorphe à la première dont les variables sont des agrégats des variables de la représentation originale. Utilisée par une métaheuristique, une Gestalt modifie implicitement l'effet des opérateurs en généralisant leur effet aux agrégats. La manipulation d'un agrégat revient à manipuler ses constituants, tous ensembles. Au delà de sa définition et de son effet, une Gestalt se construit et évolue en parallèle de l'exécution de la métaheuristique. Cette construction se base, entre autres, sur la perturbation, l'application et l'évaluation des Gestalts considérées.

Considérant son côté dual fonctionnel tel qu'introduit à la Sec. 3.2.3, une Gestalt constitue un méta-modèle de recherche. Par méta-modèle de recherche, il est entendu une définition précise permettant de construire et/ou modifier un modèle de recherche. Le modèle de recherche d'un opérateur pour une représentation donnée est le graphe de voisinage associé. Or, l'utilisation d'une Gestalt de représentation modifie ce graphe de voisinage implicitement. Il s'agit effectivement de règles génériques permettant de

modifier le modèle de recherche d'un opérateur et ce quel qu'il soit. Le processus actif de construction de Gestalt peut donc, en ce sens, être considéré comme une méta-modélisation de recherche puisqu'il s'attelle à construire un méta-modèle de recherche.

Cette méta-modélisation est de plus dynamique et en ligne. En ligne, puisqu'elle s'effectue en même temps que le processus de recherche mené par la métaheuristique exécutée. Et dynamique compte tenu du fait qu'une Gestalt peut s'avérer utile selon l'état courant de la recherche bien qu'elle ne l'ait pas été dans un état précédent. La Gestalt est dynamique et évolue selon le contexte de recherche, qui, rappelons le, n'est jamais que partiel par définition du processus d'optimisation.

3.4 Conclusion

La psychologie de la Gestalt fournit une théorie générale tentant de comprendre certains processus mentaux telle que la vision ou la résolution de problèmes comme le produit d'une systémique. Selon elle, une totalité ne peut se résumer à la simple somme de ses parties. Et les parties est plus que son individualité compte tenu de son contexte. La structuration de la totalité et des parties est alors expliquée par des lois de l'organisation ad hoc. Quant ils s'agit d'expliquer le processus cognitif de résolution de problème, trois principes fondamentaux sont dégagés : l'investigation, la perturbation et la contextualisation. L'investigation consiste en l'apprentissage et la reconnaissance de la structure d'un problème. La perturbation a pour objectif de tester la stabilité de la structure découverte et d'assurer une certaine réactivité d'apprentissage. Et la contextualisation n'est autre que l'exploitation et l'évaluation de la structure du problème apprise dans sa résolution.

Inspiré de cette théorie descriptive, un concept analogue peut être développé dans le cadre de l'optimisation combinatoire et des métaheuristiques. Pour ce faire, un parallèle est établit entre les humains résolvant un problème et les métaheuristiques traitant une instance de POC. Ce parallèle nous amène à considérer la représentation comme la perception qu'a la métaheuristique du problème. Ainsi, changer la représentation, modifie la perception de la métaheuristique et donc sa manière d'agir, puisque le graphe de voisinage résultant de la représentation est modifié en conséquence.

Une Gestalt canonique de la représentation d'une instance de POC revient alors à l'agrégation de variables en de nouvelles entités constituantes pouvant être manipulées par les opérateurs en tant que telles. Chaque opérateur est de ce fait généralisé par la délégation de son opération aux

parties de l'ensemble qu'il manipule. Ainsi, non seulement une Gestalt de représentation est une abstraction structurelle de par l'agrégation de variables, mais elle est également une abstraction fonctionnelle du processus de recherche. La construction d'une Gestalt s'articule, pour sa part, autour des trois principes que sont l'investigation, la perturbation et la contextualisation. L'investigation a pour objectif de comprendre la structure de l'instance de POC traité et la production en résultat de cet apprentissage de nouvelles Gestalts. Pour ce faire, elle se base sur les principes de perturbation et de contextualisation. La perturbation assure l'innovation et l'adaptation en introduisant explicitement de petites variations ou erreurs. Et la contextualisation met en pratique une Gestalt et évalue son utilité. Cette évaluation dirige les règles génériques de constructions mises en oeuvre par l'investigation.

Tout ceci s'inscrit dans une approche plus générale que nous appelons la méta-modélisation dynamique et en ligne. Cette approche a pour objectif d'assister en ligne une métaheuristique dans sa tâche d'optimisation. Une Gestalt de représentation peut en effet être vue comme un méta-modèle de recherche, autrement dit un cadre permettant de construire et/ou modifier le modèle de recherche qui n'est autre que le graphe de voisinage. La construction d'une Gestalt de représentation est dès lors une méta-modélisation de recherche en ligne et dynamique.

4
L'heuristique de la Gestalt

Au cours du chapitre précédent, nous avons discuté un nouveau concept algorithmique dédicacé à assister une métaheuristique dans sa tâche d'optimisation. Ce concept, inspiré de la psychologie de la Gestalt, consiste en la construction de Gestalt de représentation de telle sorte à modifier indirectement le processus de recherche de la métaheuristique. Ce concept, nous l'appelons l'heuristique de la Gestalt, sujet de la thèse. Nous nous attelons au sein de ce chapitre à la formaliser, à décrire la manière dont on peut l'appliquer à une métaheuristique, en particulier un AE, et à développer une implémentation évolutionniste pour la réaliser. Aussi, le chapitre est-il organisé en trois sections principales.

La première section est dédiée à la formalisation de l'heuristique de la Gestalt. Cette formalisation se fonde sur la définition de la représentation et du graphe du voisinage introduite au Chapitre 2. Elle se décline en trois volets permettant de définir tour à tour la Gestalt de représentation, la transformation implicite des opérateurs et l'heuristique de la Gestalt en elle-même.

Nous abordons dans la section suivante la question de l'application de l'heuristique à un AE. Nous y décrivons les modification à effectuer sur la représentation des solutions candidates et les opérateurs de l'AE afin de permettre à l'heuristique de la Gestalt à lui fournir son assistance.

Vient alors la dernière et troisième section portant sur l'implémentation évolutionniste de l'heuristique de la Gestalt. L'implémentation que nous proposons repose sur un AG dédié aux problèmes de groupements, un modèle de multi-populations, un modèle de sélection multi-niveaux et une

coévolution. Nous y décrivons exhaustivement l'algorithme résultant en passant en revue les sujets suivants : représentation, opérateurs, structure de population, évaluation, sélection et migration. Pour chacun, nous justifions nos choix et explicitons notre démarche.

En dernier lieu, nous concluons le chapitre en résumant les éléments importants abordés. Passons maintenant à la formalisation de l'heuristique de la Gestalt.

4.1 Formalisation

Partant de la discussion menée au cours du Chapitre 3, nous allons tâcher de formaliser une Gestalt transposée à l'optimisation combinatoire, et ce sous la forme d'un mécanisme, l'heuristique de la Gestalt, additionné à celui d'une métaheuristique donnée. Le formalisme présenté au cours de cette section-ci s'articulera autour des trois grands thèmes suivants : l'abstraction structurelle, l'abstraction fonctionnelle et l'heuristique de la Gestalt elle-même. L'abstraction structurelle s'attarde à la description des agrégations de variables inhérents à la Gestalt utilisée. L'abstraction fonctionnelle transpose sa version duale structurelle en un opérateur transformant adéquatement le graphe de voisinage et créant ainsi un méta-modèle de recherche. Et l'heuristique de la Gestalt, le sujet de la présente thèse, constitue le mécanisme capable de construire ces abstractions pour assister une métaheuristique dans sa tâche d'optimisation.

4.1.1 Abstraction structurelle : Gestalt description

Avant tout chose, rappelons qu'une représentation [1] d'une instance π de POC se définit selon un ensemble de n variables $x \in X$, où $n \in \mathbb{N}_0$, et dénommons cet ensemble la *description*. Comme nous l'avons discuté dans la Sec. 3.2.2, une Gestalt de la représentation d'une instance de POC agrège des variables en de nouvelles entités constituantes pouvant être manipulées par les opérateurs en tant que telles. Elle redéfinit donc une nouvelle description : la Gestalt description.

Une **Gestalt description** est une partition de l'ensemble des variables X, autrement dit un ensemble de sous-ensembles non-vides de X disjoints deux-à-deux et qui forment un recouvrement [2] de X. Chaque sous-ensemble définit une agrégation de variables et par là même une nouvelle entité constituante, indivisible et macroscopique : une **Gestalt variable**. Une Gestalt

1. La formalisation présentée dans cette section se base en grande partie sur la définition de la représentation présentée à la Sec. 2.1.2.3.
2. C'est-à-dire que leur union est égale à X.

reformule donc la représentation d'une instance de POC selon la Gestalt description associée.

Formellement, une **Gestalt description** G est définie comme un ensemble de m Gestalt variables $g \in G$, où $m \in \mathbb{N}_0$ et $m \leq n$, formant une partition de X. Chaque **Gestalt variable** $g \in G$ est un ensemble ordonné ou non de variables $x \in X$ et un sous-ensemble de X qui respecte les conditions suivantes :

- $g \neq \emptyset$,
- $\bigcup_{g \in G} g = X$, et
- $g \cap h = \emptyset$ si $g, h \in G$ et $g \neq h$.

Chaque Gestalt variable $g \in G$ est associée au domaine $\mathbb{D}_g$ comme étant le produit des domaines $\mathbb{D}_x$ associés aux variables $x \in g$: $\mathbb{D}_g = \prod_{x \in g} \mathbb{D}_x$.

Attardons nous quelque peu sur ces deux définitions. Chaque Gestalt variable est ainsi associée à l'ensemble-produit des domaines $\mathbb{D}_x$ associés aux variables qui composent cette Gestalt variable. Initialement, une solution candidate $s \in S$ est exprimée selon n composantes $c(x)$, une valeur de $\mathbb{D}_x$ associée à la variable x. Désormais, la valuation de la solution candidate s s'effectue à partir de la Gestalt description G, et non plus de la description X, vers $\bigcup_{g \in G} \mathbb{D}_g$ pour exprimer s en termes de m composantes $c_\gamma(g) \in \mathbb{D}_g$. Chaque composante $c_\gamma(g)$ est égale au tuple des composantes $c(x)$ de toutes les variables $x \in g$. Et l'ensemble $\mathcal{V}_\gamma$ des valuations c_γ est donc isomorphe à l'ensemble $\mathcal{V}$. La Gestalt d'une représentation ne produit qu'une autre représentation complète isomorphe à la première et capable de générer totalement l'espace de recherche S. Il n'est donc nullement question d'une génération partielle d'un sous-ensemble de S.

Une Gestalt de représentation construit donc un isomorphisme entre la représentation originale φ et la représentation obtenue γ, telle que $\varphi(s) = \{(x, c(x))\} \equiv \gamma(s) = \{(g, c_\gamma(g))\}$. Par abus de langage, nous nommerons la représentation γ la Gestalt puisque, φ étant connue, elle suffit à définir la transformation. Les Gestalts possibles sont nombreuses et peuvent être aisément calculées d'après le nombre de Gestalt descriptions. Il s'agit simplement du nombre de partitions d'un ensemble qui est donné par le nombre de Bell (Rota, 1964) :

$$B_n = \sum_{k=0}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) B_k \tag{4.1}$$

avec $B_0 = 1$ et $B_1 = 1$.

Dans un soucis de clarté, illustrons ces concepts par un simple exemple, une instance de POC représentée par un quintuplet de variables binaires $(x_1, x_2, x_3, x_4, x_5)$, leur domaine commun étant l'ensemble $\mathcal{D} = \{0, 1\}$. L'espace de recherche contient donc $2^5 = 32$ solutions candidates possibles et ils existent 52 Gestalts différentes possibles. Parmi elles, une possibil-

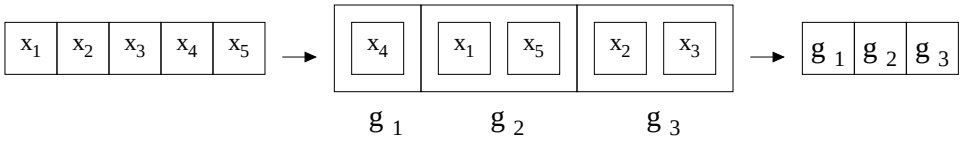


FIGURE 4.1 – Un problème représenté par le quintuplet $(x_1, x_2, x_3, x_4, x_5)$ peut être, par exemple, reformulé en terme de trois Gestalt variables $g_1 := \{x_4\}$, $g_2 := \{x_1, x_5\}$, et $g_3 := \{x_2, x_3\}$.

ité serait construite sur base de la Gestalt description $G = \{g_1, g_2, g_3\}$, où $g_1 := \{x_4\}$, $g_2 := \{x_1, x_5\}$, et $g_3 := \{x_2, x_3\}$ avec pour domaines respectifs $\{0,1\}$, $\{(0,0),(0,1),(1,0),(1,1)\}$ et $\{(0,0),(0,1),(1,0),(1,1)\}$ (voir Fig. 4.1). Il est clair que l'on régénère toutes les 32 solutions candidates.

Comme nous l'avions déjà introduit dans la Sec. 3.2.2, le rôle d'une Gestalt n'est pas de réduire l'espace de recherche à une de ses parties. Reprenant l'exemple illustré sur la Fig. 4.1, l'espace de recherche initial continue d'être représenté complètement sous la Gestalt description G. Comme nous le verrons dans la section suivante, et en particulier sur la Fig. 4.2, une Gestalt transforme le graphe de voisinage avec pour conséquence possible l'impossibilité d'atteindre une certaine zone de l'espace de recherche.

Selon une Gestalt γ, l'instance du POC considérée est reformulée en termes de Gestalt variables qui sont des entités agrégeant de plus "petites" entités. Une abstraction structurelle est ainsi construite par dessus une représentation originale. Cette abstraction structurelle en supporte une autre, mais de nature fonctionnelle : la manipulation d'une Gestalt variable se voit être reproduite à toutes ses variables constitutives. Ceci constituera le sujet de la section suivante.

4.1.2 Abstraction fonctionnelle : lentille

Une Gestalt d'une représentation change la description en agrégeant les variables en des groupuscules insécables, les Gestalt variables. Dès lors, les opérateurs de la métaheuristique ne peuvent plus travailler de la même manière. Comme nous l'avons introduit dans la Sec. 3.2.3, les opérateurs doivent désormais les manipuler en respectant leur intégrité, donnant lieu à une généralisation de l'opération à l'ensemble par sa délégation à chacun des membres de cet ensemble. La délégation est en effet le reflet de l'agrégation des composantes $c(x)$ en une composante $c_\gamma(g)$ respectant ainsi la Gestalt description définie. Cette transformation de l'opérateur et de la représentation résulte en une simplification synthétique et globale du graphe de voisinage, autrement dit une abstraction fonctionnelle du pro-

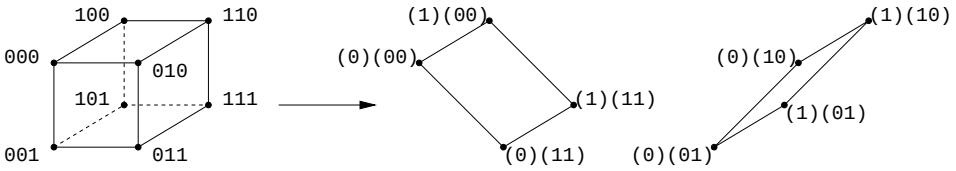


FIGURE 4.2 – La figure illustre la transformation du graphe de voisinage selon une Gestalt selon une perspective globale. Considérons une instance de problème représentée par trois variables binaires et le basculement d'un bit comme opérateur. Le graphe de voisinage original est représenté sur la gauche. Le graphe de voisinage résultant de la Gestalt appliquée est, quant à lui, représenté sur la droite.

cessus de recherche. Voyons maintenant comment formaliser cette transformation.

Reprenant le formalisme introduit dans la Sec. 2.2.3, rappelons qu'un opérateur définit pour une certaine représentation une relation de voisinage N supportée par la fonction $n : S \to 2^S$. Et cette relation engendre sur l'espace de recherche S le graphe de voisinage $v = (S, N)$. Considérant une Gestalt γ, nous avons vu dans la section précédente qu'elle permettait toujours de générer le même espace de recherche. Il convient maintenant de déterminer ce que devient N, et conséquemment v. Pour ce faire, nous allons définir un opérateur, appelé la **lentille**, appliquant la fonction n sur une nouvelle fonction n_γ représentant les nouveaux mouvements possibles de l'opérateur respectant la représentation γ.

En voici la définition. Une **lentille** est un opérateur $\mathcal{G}$ qui applique la fonction de voisinage n sur la nouvelle fonction de voisinage n_γ,

$$\mathcal{G}[n] = n_\gamma : S \to 2^S \ ,$$

tel que n_γ et sa relation associée N_γ nous donnent le voisinage de s altéré sous la représentation γ en tant que l'ensemble des solutions candidates s' pouvant être obtenu par une puissance de la relation [3] N,

$$\{s' \in S | \exists k \in \mathbb{N}_0, sN^k s' = sN_\gamma s'\} \ .$$

Il en résulte un nouveau graphe de voisinage $v_\gamma = (S, N_\gamma)$. Et considérant l'ensemble, généré par la fonction n_γ, de tous les couples de voisins $\mathcal{N}_\gamma \in S \times S$, la condition

$$|\mathcal{N}_\gamma| \leq |\mathcal{N}| \ ,$$

qui exprime la diminution éventuelle du degré de voisinage, sera respectée.

3. La puissance d'une relation correspond à sa multiple composition : xN^2z tel qu'il existe $y \neq x$ et $\neq z$ vérifiant xNy et yNz.

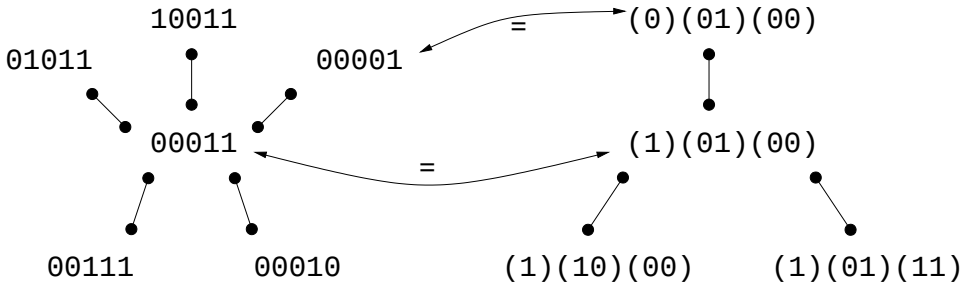


FIGURE 4.3 – La figure présente le voisinage local résultant d'un opérateur sans ou avec lentille. Considérons une instance de POC encodée par un quintuplet de cinq variables binaires et la solution candidate $(0,0,0,1,1)$. L'opérateur de basculement d'un bit peut donner cinq voisins différents tels qu'on peut l'observer sur la gauche de la figure. Considérons ensuite la Gestalt description $G = (g_1, g_2, g_3)$ avec $g_1 \coloneqq \{x_4\}$, $g_2 \coloneqq \{x_1, x_5\}$, et $g_3 \coloneqq \{x_2, x_3\}$. La solution candidate $(0,0,0,1,1)$ est alors encodée comme le triplet $([1],[0,1],[0,0])$. Et ses voisins ne sont plus qu'au nombre de trois, comme il l'est montré sur la droite de la figure. Seul un voisin original a été conservé, $(0,0,0,0,1)$, tandis que les deux autres sont nouveaux.

En transformant le graphe de voisinage, la lentille définit les nouvelles possibilités de l'opérateur, possibilités de mouvement respectant la Gestalt appliquée. La lentille transforme la perception de la métaheuristique en adaptant ses capacités de mouvement (voir Fig. 4.2, mais également Fig. 3.5). Ainsi, pour chaque solution candidate $s \in S$, et pour chacun de ses voisins $sN_\gamma s'$, il existe un et un seul pas d'opérateur qui applique s sur s' sans briser la Gestalt description définie G. La relation N_γ respecte bien la Gestalt description G. Nous voyons apparaître ici l'aspect dual de l'abstraction induite par la Gestalt, une dualité structurelle et fonctionnelle. L'ensemble des lentilles est d'ailleurs isomorphe à l'ensemble des Gestalt descriptions. Leur nombre est donc également donné par l'Equation 4.1.

A nouveau, nous nous permettons une parenthèse afin d'illustrer le concept de lentille. Pour ce faire, nous reprenons l'exemple d'instance de POC introduite lors de la section précédente. Cette instance est représentée par un quintuplet de variables binaires $(x_1, x_2, x_3, x_4, x_5)$, leur domaine commun étant l'ensemble $\mathbb{D} = \{0,1\}$. Et considérons un opérateur de basculement d'un bit. Deux solutions candidates sont voisines si et seulement si elles diffèrent en valeur d'une et une seule variable, autrement dit si la distance de Hamming qui les sépare est de un. Par exemple, les solutions candidates $(0,0,0,1,1)$ et $(0,0,0,1,0)$ sont voisines. Le graphe de voisinage correspond ainsi à un 5-cube. En appliquant une Gestalt, l'opérateur de basculement d'un bit est modifié. En particulier, lorsqu'appliqué à une Gestalt variable g, il bascule toutes les variables binaires qui la composent. Si elles sont au nombre de l, c'est donc tout un l-tuple qui est entièrement

basculé. La solution candidate résultante d'un tel mouvement serait alors distante de la solution candidate originale de l bits. Reprenons la Gestalt description utilisée dans l'exemple de la section précédente $G = \{g_1, g_2, g_3\}$, où $g_1 \coloneqq \{x_4\}$, $g_2 \coloneqq \{x_1, x_5\}$, et $g_3 \coloneqq \{x_2, x_3\}$. La lentille associée redéfinira le voisinage comme nous allons le montrer maintenant. Nos deux solutions candidates $(0,0,0,1,1)$ et $(0,0,0,1,0)$ sont respectivement représentées par $([1],[0,1],[0,0])$ et $([1],[0,0],[0,0]$. Elles ne peuvent désormais plus être voisines puisqu'il n'existe aucune transition unique permettant de passer de $g_2 = [0,1]$ à $g_2 = [0,0]$. En effet, l'application de l'opérateur sur g_2 ne peut que nous donner le couple $[1,0]$ puisque le basculement est délégué par la lentille à chacune des variables qui compose g_2. Remarquons également que la solution candidate $([1],[0,1],[0,0])$ ne possède plus que trois solutions candidates voisines comme la Fig. 4.3 nous le montre : $([1],[1,0],[0,0])$, $([1],[0,1],[1,1])$, ou $([0],[0,1],[0,0])$.

En conclusion, la lentille assure la généralisation de l'opérateur aux Gestalt variables par sa délégation. Formellement, elle revient à transformer la relation de voisinage et de ce fait le graphe de voisinage, en somme elle donne lieu à une abstraction fonctionnelle. Le premier effet visible de cette abstraction étant la réduction du degré de voisinage. Garante de l'abstraction structurelle en créant une abstraction fonctionnelle, la lentille incarne certainement au mieux l'impact d'une Gestalt sur la tâche d'optimisation que mène la métaheuristique assistée. Cependant, la lentille seule n'est rien. La Gestalt est également un processus assurant son adaptation comme nous l'avons préalablement discuté dans la Sec. 3.2.4. Nous avons désormais toutes les cartes en main pour aborder l'heuristique de la Gestalt à proprement dite.

4.1.3 L'heuristique de la Gestalt

Avant de définir formellement l'heuristique de la Gestalt, revenons sur l'aspect dynamique du processus que nous avons évoqué dans la Sec. 3.3. Pouvant s'avérer plus ou moins utile selon le contexte de recherche, une Gestalt doit être adaptée dynamiquement pour espérer maximiser son assistance de la métaheuristique utilisée. Une lentille transformera en effet le graphe de voisinage, et de ce fait le processus de recherche, en une version permettant d'assister plus ou moins la métaheuristique dans sa tâche. La qualité de l'assistance dépend ainsi de trois facteurs différents :

- l'instance de POC traitée et la représentation choisie,
- la métaheuristique utilisée en particulier les opérateurs qu'elle met en jeu, et
- l'état actuel de la recherche menée par la métaheuristique.

Attardons-nous plus précisément sur chacun de ces trois facteurs.

L'instance de POC est caractérisée par certaines propriétés telles que les dépendances entre variables. Ces propriétés peuvent s'avérer critiques dans l'efficacité d'une métaheuristique. L'exemple le plus trivial étant une métaheuristique dont les opérateurs ont été spécialisés à un POC en exploitant explicitement une propriété. Considérant une lentille, certaines de ces propriétés pourraient être oblitérées ou, dans le meilleur des cas, renforcées. Mais faut-il qu'une abstraction, quelle qu'elle soit, soit intéressante. Une instance n'arborant aucune structure globale, le cas extrême étant l'aiguille dans une botte de foin, l'abstraction ne changerait en rien cet état de fait.

Dans tous les cas, l'exploitation d'une propriété ne pourra être faite que si les opérateurs y sont sensibles. Nous retrouvons ici la difficulté de traiter séparément représentation et opérateur. Seule leur association n'a réellement de sens, puisqu'elle implique le modèle de recherche qu'est le graphe de voisinage. De ce fait, une lentille n'a de sens pour la métaheuristique que pour autant les nouvelles relations induites de voisinage aient un sens par rapport à l'instance de POC traitée et à la métaheuristique utilisée. Cette difficulté n'est pas sans rappeler le concept de Jones d'"un opérateur, un paysage d'adaptation" (Jones, 1995).

Le dernier facteur, l'état actuel de recherche, est probablement le plus spécifique à la perspective de la Gestalt. Il détermine en grande partie le contexte de la recherche, contexte à partir duquel une Gestalt prendra sens ou non. Dans notre cadre d'optimisation, la lentille octroie ou interdit l'accès à une solution candidate compte tenu de la (ou les) solution(s) candidate(s) d'origine. Et une lentille, qui, jusqu'alors, avait montré ses qualités d'assistance de la recherche, peut finalement devenir un obstacle à la progression compte tenu du nouvel état de recherche. Un exemple trivial est une lentille fournissant une abstraction de très haut niveau permettant la métaheuristique d'atteindre rapidement la zone où se trouve l'optimum global. A ce moment là, il devient impossible pour la métaheuristique d'atteindre l'optimum avec la lentille appliquée, il convient alors de la modifier afin de rétablir les relations de voisinages le permettant. Sans une étude approfondie de l'instance traitée du problème et de la métaheuristique utilisée, il sera impossible de concevoir a priori la lentille idéale compte tenu une solution candidate de départ donnée. Sans connaissance a priori, il convient donc de développer un mécanisme capable d'adapter les lentilles dynamiquement et en ligne, comme nous y étions enclins en s'inspirant de la psychologie de la Gestalt (voir Sec. 3.3). Ainsi, la métaheuristique pourra être assistée au mieux dans sa tâche selon le contexte de recherche, contexte évoluant tout au long du processus. Un tel mécanisme est proposé par l'heuristique de la Gestalt comme nous allons ci-après l'aborder.

Pour ce faire, rappelons les éléments constitutifs d'un processus de construction et d'adaptation de Gestalt introduits dans la Sec. 3.2.4 : investi-

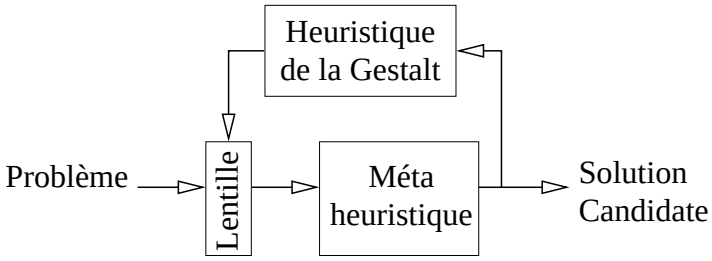


FIGURE 4.4 – Schéma bloc de l'heuristique de la Gestalt. En construisant de nouvelles lentilles qu'elle dispose entre le problème et la métaheuristique, l'heuristique de la Gestalt assiste cette dernière dans sa recherche. Cette assistance est ensuite évaluée sur base des solutions candidates obtenues afin de gouverner l'adaptation des lentilles.

gation, perturbation et contextualisation. L'investigation et la perturbation concerne la construction en elle-même, et la contextualisation l'application de la Gestalt et l'évaluation de la qualité de son assistance. Compte tenu du formalisme développé plus haut, l'heuristique de la Gestalt consiste en un mécanisme greffé sur celui de la métaheuristique. Ce mécanisme produit des lentilles qui, une fois appliquée, transforme le processus de la métaheuristique. Cette production est gouvernée entre autre par une évaluation du bénéfice apporté, bénéfice évalué sur les solutions candidates ainsi obtenues par le couplage de la métaheuristique avec l'heuristique de la Gestalt (voir Fig. 4.4).

Considérant l'instance de problème $\pi = (X, \mathcal{V}, f)$, une lentille $\mathcal{G}$ associée à une Gestalt description G, la fonction $\mathcal{M}$ d'une étape de la métaheuristique, le tuple $\mathbf{T} = (T(0), \ldots, T(t))$ où $T(0)$ est l'ensemble initial [4] de solutions candidates et $T(t)$ est l'ensemble obtenu après t itérations de $\mathcal{M}$, c'est-à-dire $T(t) = \mathcal{M}(T(t-1)) = \mathcal{M}^t(T(0))$. La **qualité d'assistance** induite par une lentille $\mathcal{G}$ permet d'évaluer la contribution de la lentille en termes de qualité de solutions trouvées ou de temps consommé. Cette qualité est calculée selon le critère $C : (\pi, \mathcal{G}, \mathcal{M}, \mathbf{T}, f(\mathbf{T}), t) \rightarrow \mathbb{R}$. L'**heuristique de la Gestalt** est alors définie comme étant la fonction $B(\mathcal{G}, C) = \mathcal{G}'$ qui maximise la qualité d'assistance donnée par le critère C.

Comme nous en avons débattu plus haut, la qualité d'assistance d'une lentille, autrement dit l'application d'une certaine Gestalt, dépend de plusieurs facteurs. L'établissement d'un modèle d'évaluation exact de cette qualité demanderait une connaissance exacte de ces facteurs et leurs impacts. Cette nécessité ne peut être remplie puisque nous considérons ne pas connaître, du moins complètement, a priori l'instance du problème. Tout au plus

4. Nous considérons ici les métaheuristiques à base de population. Le principe reste applicable aux métaheuristiques à base de parcours. Il suffit pour ce faire de considérer dans la définition une solution candidate en lieu et place d'un ensemble.

pouvons-nous seulement estimer la qualité d'assistance. Cette estimation s'incarne dans le critère de qualité d'assistance C. Bien entendu, lors de la conception de ce critère, il conviendra d'assurer un compromis entre la qualité de l'estimation et le temps de calcul nécessaire pour l'accomplir.

Après application d'une lentille et son évaluation, la contextualisation est alors pleinement réalisée. La Gestalt heuristique B peut alors, sur base des résultats obtenus, adapter, modifier, ou rechercher une ou plusieurs nouvelles lentilles afin d'améliorer son assistance. Ce processus assurera, entre autres, la perturbation des lentilles utilisées afin d'assurer une adaptation optimale. L'investigation, elle, pourra se baser sur l'une ou l'autre propriété propre à une instance de problème pour autant qu'une telle connaissance soit à disposition. Malheureusement, ce genre de cas de figure ne sera pas possible dans la majorité des cas. Aussi faudra-t-il concevoir un processus générique détaché de tout connaissance a priori.

Ce sont probablement les seul indices de réalisation de la fonction B que nous pouvons donner dès à présent. Néanmoins, nous pouvons reconnaître là une problématique d'apprentissage par renforcement (Sutton and Barto, 1998). Il s'agit en effet d'apprendre, à partir d'expériences que sont les contextualisations, ce que la métaheuristique peut ou ne peut pas faire, en terme de mouvement dans l'espace de recherche, en appliquant et modifiant des lentilles, de manière à assister son processus de recherche. Les lentilles peuvent alors être considérées comme des stratégies guidant la recherche, autrement dit l'action des opérateurs. Ceci rejoint nos commentaires sur la méta-modélisation exposés à la Sec. 3.3 : une Gestalt, réalisée en une lentille, constitue un méta-modèle de recherche modifiant le modèle de recherche, le graphe de voisinage. Ce point de vue nous permettra de choisir comment implémenter l'heuristique de la Gestalt, comme nous le verrons plus loin.

4.2 Application d'une lentille à un AE

L'heuristique de la Gestalt assiste la métaheuristique dans sa tâche en appliquant des lentilles afin de transformer son processus de recherche d'une manière ou d'une autre (voir Fig. 4.4). La lentille constitue donc l'interface d'action de l'heuristique de la Gestalt sur la métaheuristique. Il convient donc, avant tout autre considération d'implémentation, de se pencher sur les modifications qu'il est éventuellement nécessaire d'apporter à l'encodage et l'implémentation des opérateurs afin de réaliser cette interface. Tel est l'objectif de la présente section. Nous répondrons à la question, en illustrant nos propos sur l'application d'une lentille à un AE particulier, le Simple AG (SAG). Le SAG est l'un des deux AEs servant à supporter expérimentalement la thèse de l'heuristique de la Gestalt. Il s'agit d'un AG

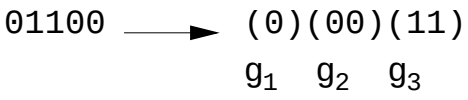


FIGURE 4.5 – La figure présente l'application de l'individu représenté par $(0,1,1,0,0)$ vers sa nouvelle représentation $([0],[0,0],[1,1])$ obtenue selon la Gestalt description $g_1 := \{x_4\}$, $g_2 := \{x_1, x_5\}$ et $g_3 := \{x_2, x_3\}$.

qui évolue des populations sans recouvrement d'individus encodés par des vecteurs de bits. Ses opérateurs sont la recombinaison à un point, la mutation à basculement et la sélection proportionnel au niveau d'adaptation (Goldberg, 1989; Mitchell, 1996) (pour de plus amples informations, voir l'Appendice A). Les modifications nécessaires à l'accueil d'une lentille seront aborder successivement selon trois thèmes : la représentation, l'évaluation des solutions candidates, et les opérateurs de recombinaison et de mutation.

4.2.1 Représentation

Précédemment, nous avons défini un Gestalt comme une représentation isomorphe à l'originale. Afin de supporter l'application de la lentille, il convient donc d'ajouter cette représentation dans l'encodage des solutions candidates. Cet ajout peut simplement s'exécuter en utilisant la Gestalt description associée à la lentille afin d'interpréter en terme de Gestalt variables.

Prenons par exemple un problème binaire dont les individus sont encodés par des vecteurs de 5 bits $x = (x_1, x_2, x_3, x_4, x_5)$ où $x_i \in \{0,1\}$ pour $i = 1, \ldots, 5$. Un exemple d'individu serait $(0,1,1,0,0)$. Considérons la Gestalt description $G = g_1, g_2, g_3$ avec $g_1 := \{x_4\}$, $g_2 := \{x_1, x_5\}$ et $g_3 := \{x_2, x_3\}$. L'individu est dès lors représenté par un vecteur de Gestalt variables $(g_1, g_2, g_3) \in \{0,1\} \times \{(0,0),(0,1),(1,0),(1,1)\}^2$, par exemple $([0],[0,0],[1,1])$ (voir Fig. 4.5). L'obtention d'une telle représentation nécessite tout au plus un adressage relatif des valeurs que prennent les variables dans une Gestalt variable.

4.2.2 Evaluation

L'application d'une lentille n'affecte nullement le processus d'évaluation d'un individu. Nonobstant le coût éventuel [5] et négligeable de l'obtention de l'individu sous sa représentation original, la fonction objective, ou d'adaptation selon l'instance de POC traitée, s'opère toujours de la même manière.

5. Généralement, ce coût est imputé lors de l'obtention de l'individu sous sa Gestalt.

4.2.3 Opérateurs

La lentille a pour effet d'altérer les opérateurs. Cette altération se résume à leur généralisation aux Gestalt variables de telle manière à ce que leur action sur une Gestalt variable soit déléguée aux variables la composant. L'implémentation d'une telle modification peut aisément se réaliser grâce au paradigme orienté objet comme nous l'exposons dans l'Appendice D.

4.2.3.1 Mutation

Une mutation applique de petites variations aux génotypes. Sous l'effet d'une lentille, la mutation opère une petite variation directement aux Gestalt variables, et de ce fait, une petite variation à toutes les variables de la Gestalt variable concernée par la mutation. Ainsi, l'impact de la variation est proportionnelle à la taille de la Gestalt variable. L'individu muté est dès lors d'autant plus éloigné que la mutation s'applique à une Gestalt variable agrégeant un grand nombre de variables.

La mutation à basculement du SAG va dès lors basculer toute une Gestalt variable, et en conséquence, toues les variables qui la composent. Par exemple, prenons l'individu représenté par $([0],[0,0],[1,1])$. Le basculement de la seconde Gestalt variable donne alors l'individu représenté par $([0],[1,1],[1,1])$ (voir Fig. 4.3).

4.2.3.2 Recombinaison

La recombinaison reproduit deux individus, les parents, ensemble afin de donner naissance à un ou deux descendants. L'idée clef de cet opérateur est l'héritage de parties chromosomiques venant des deux parents. Sous l'effet d'une lentille, la transmission de parties de chromosomes ne pourra s'effectuer qu'en respectant les Gestalt variables comme des gènes à parts entières.

Le SAG utilise la recombinaison à un point. Lorsqu'une lentille est appliquée, le sectionnement ne peut pas être effectué dans les Gestalt variables, mais uniquement entre. Le nombre de points de section est donc égale ou moire qu'en l'absence de lentille. Par exemple, avec les parents $([0],[0,0],[1,1])$ et $([1],[1,0],[0,1])$, et un point de section entre la deuxième et troisième Gestalt variable, les descendants obtenus sont $([0],[0,0],[0,1])$ et $([1],[1,0],[1,1])$ (voir Fig. 4.6).

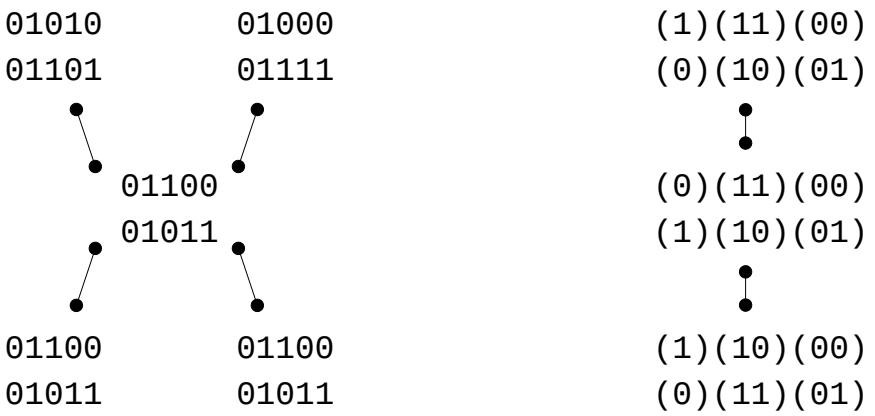


FIGURE 4.6 – Considérons un problème représenté par des vecteurs de 5 bits et deux individus $(0,1,0,1,1)$ et $(0,1,1,0,0)$. Une recombinaison à un point peut sectionner les chromosomes en quatre points différents et donner ainsi quatre couples différents de descendants, tels les individus représentés en satellite de nos deux individus parents sur la gauche de la figure. Considérons une lentille associée à la Gestalt description $G = g_1, g_2, g_3$ où $g_1 := \{x_4\}, g_2 := \{x_1, x_5\}, g_3 := \{x_2, x_3\}$. Les individus $(0,1,0,1,1) = ([0],[0,0],[1,1])$ et $(0,1,1,0,0) = ([1],[1,0],[0,1])$ peuvent, lors de leur recombinaison, produire seulement deux couples possibles de descendants, individus satellites des parents sur la droite de la figure.

4.2.4 Conclusion

L'application de la lentille ne nécessite pas de modification de la métaheuristique qui sera assistée, mais plutôt d'étendre son implémentation. L'extension se traduit par deux ajouts principaux :

- la transformation de la représentation d'une solution candidate en sa version Gestalt, et
- la généralisation des opérateurs aux Gestalt variables réalisée par la délégation aux variables composantes.

Ces ajouts ne modifient en rien l'algorithmique générale de la métaheuristique et peuvent être accompli très simplement si sa programmation est orientée objet (voir l'Appendice D).

Ceci étant discuté, seule reste la question de l'implémentation de l'heuristique de la Gestalt afin de réaliser pleinement le couplage avec une métaheuristique. Tel est le sujet de la prochaine section.

4.3 Implémentation évolutionniste de l'heuristique de la Gestalt

L'heuristique de la Gestalt ne forme pas un algorithme d'optimisation complet, mais un algorithme qui se greffe sur une métaheuristique existante afin de l'assister dans sa tâche d'optimisation. Lors de sa formalisation, nous avons vu que son processus se composait de trois parties principales : l'application d'une lentille $\mathcal{G}$, l'évaluation de son assistance grâce au critère C et la production d'une nouvelle lentille au moyen de la fonction B. Nous avons résolu la première partie lors de la section précédente. Et nous allons aborder les deux derniers points au cours de la présente section.

La définition de l'heuristique ne précise pas les détails d'implémentation. Aussi, celle-ci reste-t-elle libre et constitue un problème en soi. Certaines constatations ont cependant été faites afin de guider quelque peu notre choix. En particulier, nous avons souligné les similitudes avec les problématiques d'apprentissage par renforcement. Dans ce cadre, il les lentilles apparaissent comme des stratégies guidant la recherche, comme nous l'avons déjà expliqué. Les états sont tout simplement les différents états de recherche, c'est-à-dire la ou les solutions candidates actuelles, et les actions possibles celles déterminées par les opérateurs. Compte tenu du nombre d'états possibles, il convient de se tourner vers des techniques cherchant directement dans l'espace des stratégies, comme les AEs (Glickman and Sycara, 2001; Moriarty et al., 1999; Sutton and Barto, 1998). Et nous avons choisi effectivement ces derniers pour implémenter l'heuristique de la Gestalt. Ce choix se motive également par les deux considérations suivantes. Premièrement, les AEs utilisent généralement des opérateurs de mutation, un type d'opérateur qui fait écho à l'un des trois principes piliers de l'heuristique de la Gestalt : la perturbation. Deuxièmement, comme nous le détaillerons plus tard, certains AEs sont conçus spécialement pour traiter des problèmes de groupement. Et la problématique posée par la construction des lentilles peut se formuler en tant que tels.

Dans la suite de la section précédente, la métaheuristique assistée reste le SAG. Son couplage avec l'AE implémentant l'heuristique de la Gestalt donne lieu à un enchevêtrement d'AEs. Le SAG a pour tâche de trouver une bonne solution à un problème donné. L'AE "Gestalt" assiste le SAG dans cette tâche. L'Alg. 2 nous donne une vue d'ensemble de cet enchevêtrement (pour rappel du SAG originel voir l'Alg. 4 dans l'Appendice A).

Décrivons quelque peu l'Alg. 2. L'étape de la ligne 6 évalue la qualité d'assistance des différentes lentilles. Cette évaluation est effectuée sur base des solutions candidates trouvées après un certain nombre de génération du SAG. L'étape suivante, celle de la ligne 7, construit les nouvelles lentilles en fonction des précédentes et leurs qualités estimées respectives. L'étape

Algorithm 2 Un SAG assisté par l'AE de l'heuristique de la Gestalt

```
1: initialisation d'une population d'individus
2: initialisation d'une population de lentilles
3: application des lentilles sur les individus
4: while critère d'arrêt non satisfait do
5:    if des lentilles sont obsolètes then
6:       évaluation de la qualité des lentilles
7:       construction de nouvelles lentilles
8:       application des lentilles sur les individus
9:    end if
10:   sélection des individus pour reproduction
11:   recombinaison et production des descendants
12:   mutation de tous les descendants
13:   évaluation de la nouvelle génération
14:   replacement de l'ancienne population par la nouvelle
15: end while
```

de la ligne 8 applique les nouvelles lentilles aux solutions candidates. Ces trois étapes ne sont appliquées que lorsqu'un critère d'obsolescence des lentilles actuelles est rempli (voir la ligne 5). L'obsolescence des lentilles nécessite d'être déterminée avec soin. Elle détermine en effet la pertinence des lentilles dans le contexte de recherche courant. Finalement, les étapes des lignes 10 à 14 sont celles du SAG lui même. Et la boucle *while* se répète autant de fois que le critère d'arrêt du SAG n'est pas remplit.

4.3.1 Un AG de groupement pour évoluer les lentilles

Chaque lentille est associée bijectivement à une Gestalt description. Leurs espaces respectifs sont en effet isomorphes. Or une Gestalt description peut être formellement vue comme une partition de l'ensemble des variables. De ce fait la construction des lentilles peut être, comme déjà mentionné quelque peu avant, considérée comme un problème de groupement [6].

Partant de là, il nous est nécessaire de trouver un mécanisme capable de définir les lentilles en tant que tel. La plus simple solution se borne à générer aléatoirement les groupements et à les changer occasionnellement. Une autre consiste à optimiser les groupements de variables. Et pour ce faire, les AEs dédicacés à de tels problèmes se trouvent être un choix naturel. La première version d'AE spécialement conçu à cet effet est l'AG de Groupement (AGG), introduit par Falkenauer (1994) et exposé en détail

6. Falkenauer (1996) définit un problème de groupement comme un problème ayant pour tâche de partitionner un ensemble d'objets en une collection de sous-ensembles mutuellement disjoints, et ce selon une série de contraintes définissant les groupements valides spécifiquement au problème.

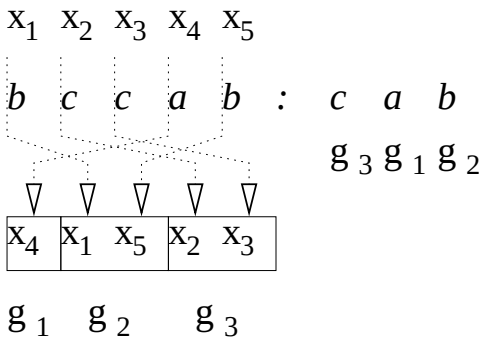


FIGURE 4.7 – La figure schématise la représentation d'une lentille en terme de groupements. Tout en haut de la figure se trouve un chromosome de cinq gènes $(x_1, x_2, x_3, x_4, x_5)$. Juste en dessous, on peut trouver la représentation d'une lentille en terme de groupements : $(b,c,c,a,b)(c,a,b)$. La partie de droite (c,a,b) définit les appellations des Gestalt variables : c pour g_3, a pour g_1 et b pour g_2. L'ordre des appellations s'avère importante lors de la recombinaison, et peut être modifiée par l'opérateur d'inversion. La partie de gauche (b,c,c,a,b) définit l'appartenance de chaque variable à chaque Gestalt variable, d'après la position des variables. Et ainsi, $g_1 = x_4$, $g_2 = x_1, x_5$ et $g_3 = x_2, x_3$.

à l'Appendice B. D'autres versions ont par après vu le jour (Chen et al., 2007; Moraglio et al., 2007; Tucker et al., 2005) afin de tenter de résoudre les problèmes de dégénérescence [7] et de redondance [8] dont souffre quelque peu la représentation qu'utilise l'AGG. Toutefois, l'AGG reste une approche efficace dans de nombreuses applications (De Lit et al., 2000; Falkenauer, 1994, 1996, 1998). Pour ces raisons et par simplicité, notre choix s'est porté et est resté sur l'AGG (Philemotte and Bersini, 2007), nous laissant le champ libre pour de futures améliorations.

4.3.2 Représentation des lentilles

La représentation de groupement proposée par Falkenauer est composée de deux parties : la partie définissant l'appartenance de chaque objet à un groupe et la partie décrivant les appellations des groupes. Cette dernière partie, bien que quelque peu redondante, est requise par la recombinaison de l'AGG (voir Appendice B).

La représentation d'une lentille consiste donc en deux parties : la partie des variables et la partie des Gestalt variables. La première est un vecteur de longueur n, c'est-à-dire le nombre de variables. La seconde partie est une liste de taille variable de m étiquettes dénommant chacune des Gestalt

7. Une représentation est dite dégénérée si plusieurs chromosomes représentent la même solution(Radcliffe and Surry, 1994).

8. Une représentation est dite redondante si une solution peut être uniquement déterminée à partir d'un sous-ensemble de gènes de son chromosome (Radcliffe and Surry, 1994).

variables. Pour chaque variable, une étiquette lui est assignée fonction de sa place ordinale [9] et désignant ainsi son appartenance à une Gestalt variable. Par exemple, prenons une instance de problème représentée par un vecteur de cinq bits $(x_1, x_2, x_3, x_4, x_5)$. Les Gestalt variables, quant à elles, sont respectivement étiquetées a, b et c. Une possible lentille serait représentée par :

$$bccab : cab\ .$$

La Gestalt description associée est alors $g_1 = x_4$, $g_2 = x_1, x_5$ et $g_3 = x_2, x_3$.

4.3.3 Opérateurs évolutionnistes

L'AGG de Falkenauer fonctionne sur base d'opérateurs spécialement conçu pour la représentation en terme de groupements. Nous proposons de rappeler, ci-dessous, l'essentiel du concept derrière chacun de ces opérateurs : mutation, inversion et recombinaison (Falkenauer, 1998) (voir Appendice B).

4.3.3.1 Mutation

Trois types de mutation sont utilisés :
- la réassignation aléatoire d'une variable à une Gestalt variable,
- la séparation d'une Gestalt variable en deux, et
- la fusion de deux Gestalt variables en une seules.

Pour chacun des types, une probabilité détermine l'application effective de la mutation (dans l'ordre $P_{R\mu}$, $P_{S\mu}$ et $P_{M\mu}$).

4.3.3.2 Inversion

Avec une probabilité P_I, l'inversion s'opère seulement sur la partie Gestalt. Deux positions sont tirées aléatoirement, et, entre elles, l'ordre des étiquettes est inversé. Cet opérateur ne change pas la définition de la lentille. Mais il joue un rôle crucial pour l'opérateur de recombinaison.

4.3.3.3 Recombinaison

La recombinaison est un opérateur complexe nécessitant plusieurs étapes algorithmiques successives. L'idée principale réside dans l'échange du matériel génétique au niveau des étiquettes de groupe (dans notre cas, de Gestalt variables). L'algorithme [10] applique une recombinaison à deux points sur

9. Dans la terminologie propre aux AEs, cette position est appelé le locus.
10. Pour une définition formelle de la recombinaison de groupement, le lecteur intéressé pourra consulter Tucker et al. (2005).

la partie de groupe des parents et échange les appartenances selon les étiquettes échangées (Falkenauer, 1998).

4.3.4 Structure de population

D'après la définition de l'heuristique de la Gestalt, une lentille est d'autant récompensée qu'elle assiste la tâche de la métaheuristique, ici le SAG. Ce dernier évoluant une population, la qualité de l'assistance dépends de l'évolution de toute la population. C'est pourquoi l'application d'une lentille s'effectue sur tous les individus de la population. La restriction à un seul individu n'aurait en effet aucun sens puisqu'il s'agit d'évaluer à quel point la lentille aide le SAG et non un unique individu.

Lorsqu'elle est appliquée à un SAG, la lentille est donc associée fortement à la population d'individus. Et tout comme le SAG, l'AGG est une métaheuristique à base de population. Il évolue donc une population de lentilles. Deux possibilités permettent de remplir ces contraintes :

- les lentilles sont appliquées successivement à la même population d'individus, ou
- les lentilles sont appliquées parallèlement à différentes populations d'individus.

La seconde possibilité n'est pas sans rappeler une structure de multi-îles. Ce type de structure offre d'intéressantes propriétés comme la diversification ou la facilité de parallélisation (Skolicki, 2007). Chaque île étant alors associée à une lentille, la représentation de leurs individus se voient êtres transformés en conséquence. On obtient ici une hétérégéonité de représentation entre les différentes îles (Skolicki and Jong, 2004; Skolicki, 2007). Dans le contexte de la Gestalt, cette perspective prend particulièrement sens. Chaque population peut être considérée comme le contexte et l'expérience attachée à une Gestalt particulière, évoluant avec cette dernière. Pour toutes ces raisons, nous avons opté pour cette structure de multi-îles. A remarquer que dans de précédents travaux, c'est la première perspective qui fut choisie (Bersini, 2004; Philemotte and Bersini, 2005a,b)

Considérons une population P de r lentilles $\mathcal{G}_i$. La population d'individus P est ainsi divisées en r îles $P_{\mathcal{G}_i}$, $i = 1, \ldots, r$, de même taille fixe ρ, et qui sont associées uniquement à chaque lentille. Chaque lentille s'applique sur l'île correspondante et altère toute opération sur ses individus (voir Fig. 4.8).

4.3.5 Evaluation des lentilles

Wiegand (2003) définit un *algorithme coévolutionniste* comme un AE employant une mesure interne et subjective pour l'évaluation de l'adaptation.

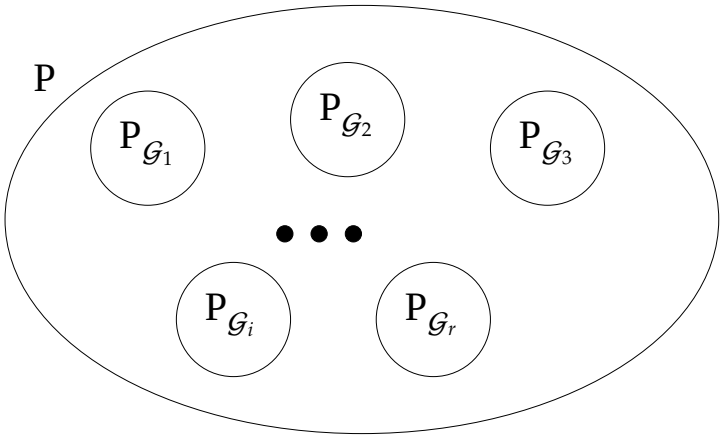


FIGURE 4.8 – Considérons une population de r lentilles $\mathcal{G}_i$. Chaque lentille est associée à une île $P_{\mathcal{G}_i}$ de taille ρ. Tous les individus qui appartiennent à la population $P_{\mathcal{G}_i}$ sont alors manipulés "au travers" de la lentille $\mathcal{G}_i$.

L'évaluation des lentilles est de ce type. Chaque lentille est en effet évaluée sur base de sa qualité d'assistance de l'évolution de son île. L'évolution des lentilles est une coévolution de différentes îles.

Mais que peut être une bonne mesure de la qualité d'une lentille ? Cette qualité d'assistance dépend de :

- les niveaux d'adaptation des individus, et
- les gradients de ces mêmes niveaux.

Considérant une certaine lentille, la première indique à quel point les résultats obtenus l'ont été grâce à cette lentille. La deuxième mesure une variation signée indiquant l'impact de la lentille sur la recherche. Une bonne lentille aide en effet à découvrir rapidement de bons individus. Ceci nous amène à définir différentes fonctions d'adaptation $g \geq 0$ de lentille $\mathcal{G}_i$ selon un seul critère :

- la qualité de l'objectif atteint :
  - le meilleur individu trouvé

$$g_{best}(\mathcal{G}_i, t) = \text{best}\left(\hat{f}(x) | x \in I_{i,t}\right) ,$$

    où best est la fonction max (min) pour un problème de maximisation (respectivement minimisation) et $I_{i,t}$ est la t-ème génération de l'île I_i après l'application de la lentille $\mathcal{G}_i$,
  - la moyenne des individus d'une île

$$g_{avg}(\mathcal{G}_i, t) = \frac{1}{\rho} \sum_{x \in I_{i,t}} \hat{f}(x) ,$$

– la moyenne pondérée par le meilleur (MPM)

$$g_{MPM}(\mathcal{G}_i, t) = \sqrt{g_{best}(\mathcal{G}_i)\, g_{avg}(\mathcal{G}_i)}$$

– la qualité de la vitesse de recherche (considérant t' avant t) :
  – la variation du meilleur individu atteint

$$g_{\nabla best}(\mathcal{G}_i) = g_{best}(\mathcal{G}_i, t) - g_{best}(\mathcal{G}_i, t')$$

  – la variation de la moyenne des individus d'une île

$$g_{\nabla avg}(\mathcal{G}_i) = g_{avg}(\mathcal{G}_i, t) - g_{avg}(\mathcal{G}_i, t')$$

  – la variation de la MPM

$$g_{\nabla MPM}(\mathcal{G}_i) = g_{MPM}(\mathcal{G}_i, t) - g_{MPM}(\mathcal{G}_i, t')$$

Ajoutons quelques mots à propos de la MPM sur laquelle nous jetterons notre dévolu. Evaluer une lentille selon les seules élites produites risque de renforcer la force de sélection, et, de ce fait, de prématurer la convergence. Par contre, une évaluation dirigée par la qualité moyenne de la population associée risque de trop affaiblir la force de sélection. La MPM permet d'avoir une mesure combinée de la qualité de l'ensemble et de son élite, et ainsi de balancer la force de sélection à bon escient.

4.3.6 Sélection des lentilles

L'enchevêtrement du SAG et de l'AGG donne lieu à une double processus de sélection compatible avec le modèle de sélection multi-niveau (Lenaerts, 2003). Les individus d'une même île coopère durant l'évaluation de la lentille qui leur attachée. Cette coopération augmente la probabilité de sélection de leur lentille. Si la lentille est sélectionné, l'île associée est promue dans la population globale. Il a été montré que cette sélection explicite de groupe augmente le comportement de coopération au sein de chaque île (Lenaerts, 2003).

Pour l'AGG, Falkenauer (1998) suggère l'utilisation d'une variante bruitée de la sélection par tournoi. Evidemment, il n'est pas impossible d'utiliser d'autres opérateurs de sélection. Ainsi, avons nous utilisé, lors de travaux précédents (Philemotte and Bersini, 2007), une simple sélection par tournoi. Cette fois nous utiliserons l'Echantillonnage Universel Stochastique (EUS) (Baker, 1987). De cette manière, nous évitons le biais de sélection et nous respectons le modèle de sélection multi-niveau qui nécessite son usage au niveau des îles (Lenaerts, 2003).

4.3.7 Migration des individus

Utilisant une structure en multi-île, chaque lentille est appliquée à une unique île. Lorsque les lentilles évoluent, de nouvelles générations sont créées au moyen de diverses opérations. Mais qu'en est-il des îles ? Les îles sont-elles complètement réinitialisées ? Ou sont-elles gardées sans aucun changement ? Puisque nous interprétons ces associations entre lentilles et îles comme étant leur contexte et expérience, les îles des parents sont transmises aux lentilles descendants obtenus suite à une recombinaison. Puisque les génotypes parentaux ont été mélangé, il est naturel d'en faire de même avec les îles. C'est ici qu'intervient l'opérateur de **migration** proposé dans le modèle des multi-île (Bessaou et al., 2000; Gordon et al., 1992; Grosso, 1985; Martin et al., 1997; Skolicki, 2007). La migration échange de l'information entre îles, autrement dit des individus.

La migration est caractérisée par la taille ρ_m, la fréquence t_m et la stratégie (Skolicki, 2007). Skolicki (2007) a montré que la diversité inter-île peut être maintenue en configurant une petite taille, une basse fréquence, une topologie éparse et une stratégie faible. La migration est implémentée en prenant compte cette conclusion. La topologie dépend de la sélection des lentilles. La taille ρ_m sera une petite fraction de la taille de l'île ρ. La fréquence sera basse. La stratégie de sélection d'émigrants et d'immigrants respecte la philosophie du modèle de sélection multi-niveau. Les émigrants sont sélectionnés par EUS et l'immigration est effectuée en remplaçant aléatoirement les individus, excepté les élites.

4.3.8 L'algorithme complet

Dans les sous-sections précédentes, nous avons décrits et argués l'implémentation de l'heuristique de la Gestalt en AGG. Ce qui nous donne en résumé l'Alg. 3.

4.4 Conclusion

Une Gestalt de représentation reformule une instance de POC selon une Gestalt description G définie comme une partition de l'ensemble des variables X, contenant ainsi des sous-ensemble g de variables appelées Gestalt variables. En conséquence, les solutions candidates sont reformulées selon une nouvelle représentation $\gamma : S \to \mathcal{V}_\gamma$ fait correspondre toute solution candidate $s \in S$ à un m-uplet de valeur $c_\gamma(g) \in \mathbb{D}_g = \prod_{x \in g} \mathbb{D}_x$ pour chaque Gestalt variable $g \in G$.

Algorithm 3 Un AGG pour l'heuristique de la Gestalt assistant un SAG

1: $t = 0$
2: initialise la population de $\mathcal{G}_i$
3: **for all** lentilles **do**
4: initialise son île $P_{\mathcal{G}_i}$
5: **end for**
6: **while** critère d'arrêt non satisfait **do**
7: **if** $t_m = t$ **then**
8: évalue chaque $\mathcal{G}_i$
9: copie les élites dans la prochaine génération
10: sélectionne des couples (i, j) de parents avec EUS
11: **for all** couples sélectionnés de $\mathcal{G}_{i,j}$ **do**
12: recombine et produit deux descendants
13: mute et inverse
14: associe $P_{\mathcal{G}_{i,j}}$ à la progéniture
15: migre des solutions entre $P_{\mathcal{G}_{i,j}}$
16: **end for**
17: **end if**
18: effectue une génération de SAG sur chaque $P_{\mathcal{G}_i}$
19: $t = t + 1$
20: **end while**

Ce changement de représentation a pour effet de modifier les opérateurs appliqués. Ce changement s'incarne dans la lentille associée qui est un opérateur mathématique $\mathcal{G}$ appliquant la fonction de voisinage n, résultant de l'opérateur et de la représentation initiale, sur une nouvelle fonction de voisinage n_γ. Il en résulte un nouveau graphe de voisinage v_γ dont le degré de voisinage pourrait être diminué. De ce fait, la lentille assure la généralisation de l'opérateur aux Gestalt variables par sa délégation.

L'heuristique de la Gestalt est responsable de la construction des nouvelles lentilles. Elle peut être définie comme la fonction $B(\mathcal{G}, C) = \mathcal{G}'$ qui maximise la qualité d'assistance donnée par le critère C. Après application d'une lentille $\mathcal{G}$ et son évaluation selon C, la contextualisation est réalisée. L'investigation et la perturbation sont quant elles réalisée au sein de la fonction B.

L'assistance d'un AE par l'heuristique de la Gestalt n'en nécessite pas la modification, mais plutôt l'extension de son implémentation. Cette extension permettra la transformation de la représentation originale en sa version Gestalt et la généralisation des opérateurs aux Gestalt variables par la délégation aux variables composantes.

Assistant un SAG, l'implémentation évolutionniste de l'heuristique de la Gestalt, autrement dit de la fonction B et du critère C, repose sur l'utilisation d'un AGG pour évoluer les lentilles. Ces dernières, ou plutôt leurs Gestalt

descriptions associées, sont alors représentées au moyen d'une représentation de groupement. La population du SAG se voit ensuite structurée en sous-populations. Sur chacune d'elles, il est appliqué une lentille. L'évaluation d'une lentille se fait alors sur base de la qualité des solutions trouvées au sein de sa sous-population associée. La sélection des lentilles implique celle de sa sous-population associée. De ce fait, les solutions candidates se retrouvent être sélectionnées selon plusieurs niveaux. Finalement, lors de la recombinaison de deux lentilles, une migration de solutions candidates est effectuées entre leurs sous-populations associées.

5
Expériences : études et résultats

Au cours du chapitre précédent, nous avons décrit comment appliquer une lentille et l'implémentation évolutionniste de l'heuristique de la Gestalt. Nous allons maintenant décrire et discuter quatre études expérimentales menées à diverses fins, chaque étude étant présentée dans une section ad hoc.

En préambule, nous rassemblons quelques considérations préliminaires et communes aux quatre expériences au sein d'une première section. Ces considérations concernent le paramétrage automatique des algorithmes utilisés, la manière dont nous mesurons le temps et le critère d'arrêt basé sur cette mesure, et la manière dont nous mesurons l'efficacité d'un algorithme.

Exposée dans la seconde section, la première expérience consiste en une preuve du concept de l'impact effectif d'une lentille sur l'optimisation menée par une SAG. Le problème traité est structuré et permet de concevoir des lentilles a priori. Ces lentilles restent statiques durant tout le processus de recherche.

Dans la troisième section, nous présentons la deuxième expérience, également une preuve de concept. L'idée est d'éprouver l'utilité d'un mécanisme élaborant en ligne des lentilles pour assister un SAG traitant un problème complexe et d'une riche structure. Afin de se concentrer sur l'objectif, l'heuristique de la Gestalt est expressément utilisée sous une forme simplifiée et spécialisée.

La troisième expérience s'attelle, au sein de la quatrième section, à étudier l'implémentation de l'heuristique de la Gestalt que nous avons détaillé au Chapitre 4. En particulier, nous nous y assurons que l'évolu-

tion des lentilles au moyen d'un AGG contribue positivement à l'assistance prodiguée à un SAG traitant des problèmes de diverses tailles et difficultés. Il y est également étudié l'évolution de la taille moyenne des Gestalt variables des meilleures lentilles trouvées.

Enfin, la dernière expérience se concentre sur une mise en situation pratique de l'heuristique de la Gestalt. L'expérience illustre ainsi la démarche ad hoc à adopter dans un autre cas de figure : autre problème, autre représentation et autre AG. Adoptant une attitude réaliste, nous montrons quelles modifications sont à opérer et les éventuelles spécialisations qui peuvent être introduites.

Nous terminons le chapitres en rappelant les conclusions de chaque expérience et en les interprétant dans leur ensemble. Et nous pouvons commencer ce long chapitre avec les quelques considérations préliminaires .

5.1 Quelques considérations préliminaires

Avant d'aborder le sujet central de ce chapitre, les expériences, nous exposons au sein de cette section quelques considérations préliminaires abordant divers sujets récurrents dans l'élaboration ou la description des expériences.

5.1.1 Paramétrage des algorithmes utilisés

Les AGs utilisés dans les expériences présentées nécessitent une configuration soignée de leurs paramètres afin de traiter les problèmes fournis le plus efficacement possible. Dans la mesure du possible, cette configuration sera effectuée au moyen de l'algorithme F-Race de Birattari et al. (2002) jusqu'à discrimination complète de la meilleure configuration. Dans tous les cas, la modalité de paramétrage sera précisée.

L'algorithme F-Race [1] se base sur l'évaluation systématique d'un jeu de configurations pour traiter différentes instances du problème. Au fur et à mesure des instances, les configurations les moins bonnes sont retirées et ne sont plus évaluées par la suite. Ce retrait s'effectue en deux étapes. Dans un premier temps, un test statistique de Friedman permet de vérifier si toutes les configurations offrent les mêmes performances. Dans un second temps, si cette hypothèse est vérifiée, un nouveau cycle commence. Dans le cas contraire, les configurations sont alors toutes comparées statistiquement avec la meilleure configuration. Et les configurations qui sont significativement les plus mauvaises sont retirées de l'ensemble des configurations à évaluer.

1. Une librairie R pour F-Race est disponible à l'URL
http://cran.r-project.org/src/contrib/Descriptions/race.html.

Et ainsi de suite, jusqu'à atteindre un nombre maximum d'évaluations ou un nombre donné de configurations.

5.1.2 Mesure du temps et critère d'arrêt

Les expériences présentées au sein de ce chapitre mettent en jeu un certain nombre d'AGs. Ces algorithmes sont capables de fournir une réponse, c'est-à-dire une ou plusieurs solutions candidates, dont la qualité évolue fonction du temps. Il est donc nécessaire de préciser la mesure du temps qui est utilisée dans la présentation des résultats.

Sauf mention contraire, le temps est mesuré en secondes de temps processeur [2] réel consommé par le programme exécutant l'AE correspondant.

Le critère d'arrêt des AGs se base d'ailleurs sur cette mesure de temps. Ainsi, sera-t-il octroyé une certaine quantité de temps processeur au-delà de laquelle l'algorithme ne pourra plus fournir aucune réponse. De cette manière, les algorithmes peuvent être comparés compte tenu de la réalité de leurs implémentations et des éventuels coûts qu'elles comportent.

5.1.3 Mesure d'efficacité

Afin de pouvoir comparer les différents algorithmes utilisés lors des diverses expériences, il est nécessaire de mesurer l'efficacité de ceux-ci. La mesure consiste en l'expression de l'évolution de la qualité des réponses en fonction du temps, le temps étant, comme expliqué ci-avant, généralement mesuré en temps processeur réel. La qualité est, quant à elle, exprimée par biais de statistiques non-paramétriques (telles que la médiane, les quartiles et les déciles) de l'erreur relative $\Delta f(s)$ de la meilleure solution candidate s par rapport à la valeur objective de l'optimum global $f(s^*)$:

$$\Delta f(s) = \frac{\left| f(s) - f(s^*) \right|}{f(s^*)} \, .$$

Ainsi, si s correspond à l'optimum s^*, l'erreur relative est nulle. Dès lors, tout problème peut être considéré comme un problème de minimisation.

Ces quelques considérations étant, nous pouvons désormais aborder les expériences en tant que telles.

2. L'ordinateur utilisé fonctionne avec un processeur AMD Athlon XP 2400+ 2 GHz couplé à 1,5 GB de mémoire RAM. Le système d'exploitation est basé sur la distribution Gentoo GNU/Linux (noyau GNU/Linux 2.6.23).

5.2 Une lentille statique et conçue a priori

Dans un premier temps, nous proposons, au sein de cette section, d'éprouver le simple concept de lentille, en particulier de son impact sur la recherche menée par un AE. Pour ce faire, les performances d'un AE seront comparées avec celles obtenues sous l'assistance d'une seule lentille statique et conçue a priori. De cette manière, le concept de la Gestalt peut être testé dans son plus simple appareil, la lentille. Ainsi, la lentille n'évoluera-t-elle pas au courant de l'optimisation et sera-t-elle construite une fois pour toute sur base de la connaissance du problème traité, problème suggérant naturellement et a priori l'agrégation de variables. Tel est le fondement de l'expérience présentée dans cette section.

Plus concrètement, l'AE assisté par les lentilles est un SAG, AG générique, simple de conception et probablement l'un des plus étudiés. Le SAG travaillant sur des représentations binaires, le problème traité doit pouvoir être représenté de la sorte. N'oubliant pas qu'il nous faut un problème présentant une structure favorisant naturellement l'agrégation de variables, le Challenge de la Route Royale (CRR) rencontre parfaitement cette contrainte. Compte tenu de l'instance de CRR précisément traitée, trois différentes lentilles sont conçues avant optimisation et leur qualité d'assistance est comparée en termes d'erreur relative par rapport à l'optimum. Mais avant de présenter et discuter les résultats obtenus, présentons plus avant les différents composants de cette expérience.

5.2.1 CRR

Le CRR, présenté par Holland (1993) à l'ICGA'93 (Culberson, 1993; Jones, 1994), fut conçu afin de tester l'hypothèse[3] des blocs constitutifs (Goldberg, 1989; Holland, 1975). Rappelons qu'un bloc constitutif est un groupe d'allèles dont les positions sont rapprochées (Holland, 1975).

Ce problème binaire hautement trompeur et démontrant une structure hiérarchique peut être formellement décrit comme la recherche du vecteur de bits $x = (x_1, \ldots, x_n)$, où $x_i \in \{0,1\}$, qui maximise la fonction objective f_{CRR} donnée par :

$$f_{\text{CRR}}(x) = \sum_{j=1}^{k+1} B(j) + \sum_{i=1}^{1+2^k} P(i) \,,$$

3. Le lecteur intéressé trouvera un complément d'information à propos de cette hypothèse à la Sec. 6.4.2.

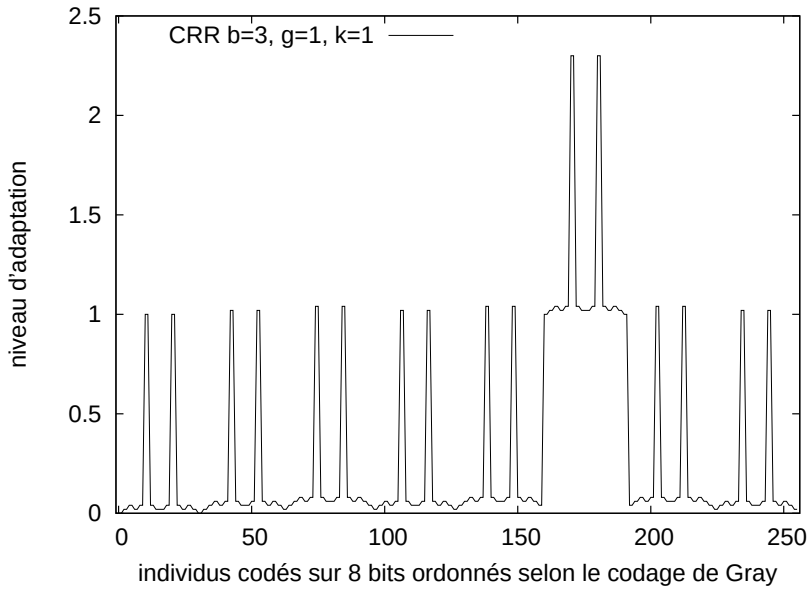


FIGURE 5.1 – Ce graphe représente le paysage d'adaptation de l'instance de CRR où $b = 3$, $g = 1$, $k = 1$, $u^* = 1$, $u = 0.3$, $v = 0.02$ et $m^* = 2$. En abscisse, les solutions candidates sont représentées et classées sous la forme d'un entier obtenu par le codage de Gray.

où $j = 1, \ldots, k+1$ indexe les niveaux de la hiérarchie, i les schemata [4] ciblés, B la fonction de bonus et P la fonction de participation. Il y a $n(j) = 2^{k-j+1}$ schemata ciblés au niveau j, composé de paires adjacentes de schemata du niveau inférieur. Chaque schema ciblé est défini par b loci à une distance de g bits de chaque schema contigu. La longueur n du vecteur se calcule simplement par $(b+g)2^k$.

La fonction de bonus B récompense positivement le vecteur x à un niveau j donné fonction du nombre de blocs corrects :

$$B(j) = \begin{cases} u^* + (n(j)-1)u \text{ , if } n(j) > 0 \\ 0 \text{ , if } n(j) = 0, \end{cases}$$

où u^* est le bonus au premier schema ciblé, et u le bonus aux schemata restants.

4. Le lecteur intéressé trouvera un complément d'information à propos du concept de schema à la Sec. 6.4.1.

La fonction de participation P calcule la contribution au score global des $m(i)$ allèles correctes [5] dans le schema ciblé i du plus bas niveau :

$$P(i) = \begin{cases} m(i)v \text{ , if } m(i) < m^* + 1 \\ -(m(i) - m^*)v \text{ , if } m* < m(i) < b \\ 0 \text{ , otherwise,} \end{cases}$$

où $m(i)$ corresponds au nombre de bits **1** dans le schema i, m^* est le nombre minimal requis de bits pour obtenir une récompense, et y la récompense/pénalité par bit.

La Figure 5.1 présente le paysage d'adaptation d'une instance de CRR où $b = 3$, $g = 1$, $k = 1$, $u^* = 1$, $u = 0.3$, $v = 0.02$ et $m^* = 2$. On y remarquera plusieurs optima locaux et deux optima globaux. La structure est relativement accidentée, signe de la récompense aux blocs correctement constitués.

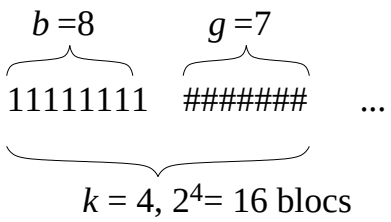


FIGURE 5.2 – La figure représente les vecteurs de bits constituant les maxima de la fonction f_{CRR}. Ils se composent de seize blocs de huit bits **1**, séparés par des bloc de sept bits de valeur quelconque, valeur quelconque représentée par le symbole #.

Dans l'expérience présentée ci-dessous, le problème CRR sera paramétré selon configuration originale d'Holland (1993) : $n = 240$, $b = 8$, $k = 4$, $g = 7$, $u^* = 1$, $u = 0.3$, $v = 0.02$ et $m^* = 4$. Dans ce cas, le maximum de la fonction f_{CRR} est de 12.8, et obtenu pour les vecteurs de bits (voir la Fig. 5.2) composés de seize blocs de huit bits **1**, chacun étant séparé par un bloc de sept bits quelconques que l'on nomme intron. Les optima sont donc au nombre de $2^{16\times7} \approx 5,2\,10^{33}$.

5.2.2 SAG et son paramétrage

Le SAG [6] utilisé, rappelons-le, consiste en un AG évoluant une population sans recouvrement initialisée aléatoirement dont les individus sont représentés sous forme de vecteur de bits, au moyen de la mutation à basculement, de la recombinaison à un point et de la sélection proportionnelle

5. Un allèle correct est communément considérée comme le bit **1**.
6. Pour plus de détails à ce sujet, le lecteur est invité à consulter l'Appendice A.

Paramètre	Valeurs possibles	Meilleure valeur
N_{pop}	{100, 250, 500, 750, 1000, 1250, 1500}	1000
P_μ	{0, 1/480, 1/240, 1/120, 1/60}	1/240
P_X	{0;0,1;0,2;0,3;0,4;0,5;0,6;0,7;0,8;0,9;1}	0,4

TABLE 5.1 – Le tableau présente les différents ensembles de valeurs pour les paramètres de taille de population N_{pop}, de probabilité de mutation P_μ et de probabilité de recombinaison P_X. Leur arrangement forme une configuration du SAG afin de traiter l'instance de CRR. Chaque configuration se voit accorder trente secondes de temps processeur.

EUS. Les paramètres de configuration du SAG sont la taille de population N_{pop}, la probabilité de mutation P_μ et la probabilité de croisement P_X.

Le SAG est un AG simple de conception et générique. Ses opérateurs ne sont en effet nullement spécifique à un problème. En somme, on peut voir en le SAG, l'algorithme canonique s'inscrivant dans le scénario de la boîte noire. C'est ce qui motive notre choix. Les lentilles n'ont de sens que lorqu'elles ont pour objectif d'assister un algorithme qui n'est pas spécialement conçu pour le problème traité, au risque de faire double emploi.

Le SAG est paramétré automatiquement, pour l'instance de CRR précitée, à l'aide de l'algorithme F-Race, présenté plus tôt dans ce chapitre. Un temps processeur de trente secondes est accordé pour chaque configuration testée. Au total, trois cent quatre-vingt-cinq configurations différentes sont testées et produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.1. La meilleure configuration obtenue est une population de 1000 individus, une probabilité de mutation de 1/240, et une probabilité de recombinaison de 0,4. Remarquons que la probabilité de recombinaison est assez faible. Ainsi, en moyenne, moins de la moitié des individus sélectionnés se recombineront. Par contre, tous subiront en moyenne au moins une mutation. Selon ce paramétrage, la mutation est donc le mécanisme de recherche dominant.

5.2.3 La conception a priori des lentilles

Nous allons concevoir quatre lentilles différentes, représentant chacune une Gestalt description différente. La conception est dirigée par la structure de l'instance du CRR. Rappelons que le CRR récompense les blocs selon leur bonne constitution (voir la Fig. 5.2), les bits des introns pour leur part n'ayant aucun impact.

La première lentille est construite de telle manière à percevoir tous les détails du problème. En somme, son application n'affecte nullement le SAG dans sa tâche d'optimisation, comme si aucune lentille n'avait été appliquée. Elle constitue en quelque sorte notre zéro de référence.

Pour la deuxième lentille, nous nous basons sur notre connaissance de l'inutilité des introns : la valeur obtenue par la fonction f_{CRR} n'est pas affectée par la valeur de chacun des bits appartenant aux introns. Les grouper en une seule Gestalt variable réduit clairement le degré de connexion du graphe de voisinage en réduisant toutes les possibilités d'opération à une seule : une opération sur la Gestalt variable des introns. Quant aux autres bits, ceux des blocs "utiles", ils sont toujours considérés individuellement.

La troisième lentille est construite sur base de la deuxième. Les introns y restent agrégés en une seule Gestalt variable. Mais, les bits adjacents des blocs "utiles" sont agrégés deux par deux. De même, pour la quatrième lentille, mais quatre par quatre.

En résumé, si nous notons b le nombre de bits adjacents d'un même bloc "utile" constitutif agrégés en une Gestalt variable, et g le nombre de bits des introns en une Gestalt variable, les quatre lentilles sus-décrites peuvent être définies respectivement par les quatre couples (b, g) suivants : $\mathcal{G}_1 = (1, 1)$, $\mathcal{G}_2 = (1, 112)$, $\mathcal{G}_3 = (2, 112)$ et $\mathcal{G}_4 = (4, 112)$.

5.2.4 Conditions expérimentales

Afin d'étudier expérimentalement l'assistance prodiguée au SAG par les différentes lentilles, l'évolution de la meilleure solution candidate est mesurée en fonction du temps processeur. Pour chaque lentille, cinq cents essais sont exécutés, chaque essai se voyant octroyé un maximum de dix secondes de temps processeur.

5.2.5 Résultats et discussion

La Figure 5.3 résume les résultats obtenus et présente la comparaison de l'assistance du SAG par nos quatre lentilles pour un temps processeur inférieur à cinq secondes (de gauche à droite, $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3$ et $\mathcal{G}_4$). Le résumé s'effectue sous la forme de boîtes à moustaches décrivant la médiane, les quartiles et les déciles de l'erreur relative Δf de la meilleure solution candidate trouvée pour chacun des cinq cents essais et fonction du temps processeur. La synthèse des résultats des tests statistiques est présentée, quant à elle, au Tableau 5.2.

Comme nous pouvons l'observer sur la Fig. 5.3 et conclure selon le Tableau 5.2, dans les premiers instants de la recherche (jusqu'à la cinquième seconde), correspondant à plusieurs dizaines de générations, l'agrégation seule des bits des introns suffit à accélérer la recherche. A remarquer également l'effet positif de l'adéquation de la lentille à la structure du CRR lors de la première seconde de recherche. Cependant, plus l'adéquation est forte

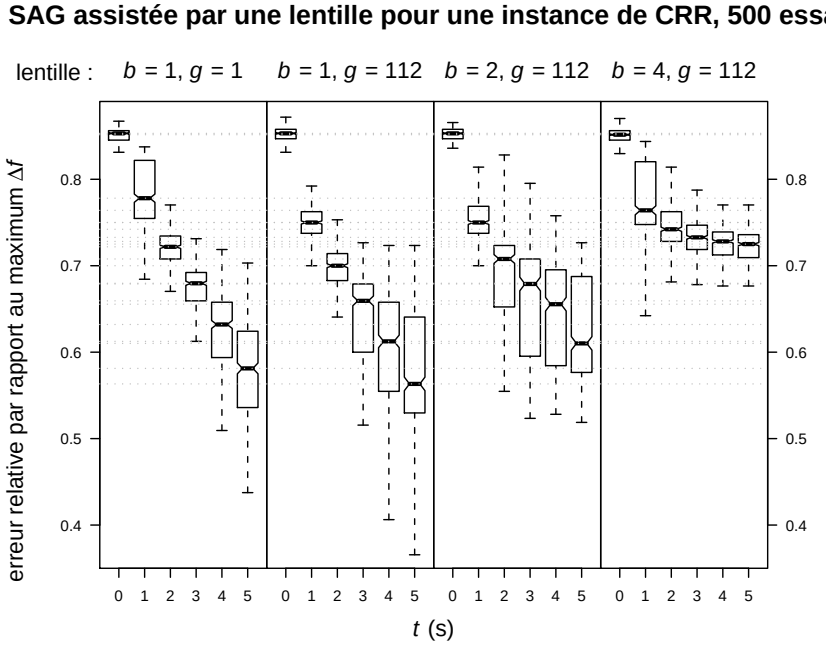


FIGURE 5.3 – Sur cette figure, les assistances du SAG par quatre différentes lentilles statiques et conçues a priori sont comparées. Pour chacune des lentilles, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates sur cinq cents essais est représentée sous forme de boîte à moustaches en fonction du temps processeur. Sur le graphique de gauche, l'évolution de la lentille référence ($b = 1$, $g = 1$) est représentée, lentille n'altérant en rien le fonctionnement du SAG. Sur les trois graphiques à sa droite, il s'agit de l'évolution des lentilles agrégeant tous les bits des introns en une seule Gestalt variable, et les bits adjacents d'un bloc "utile" constitutif successivement par un ($b = 1$, $g = 112$), par deux ($b = 2$, $g = 112$) et par quatre ($b = 4$, $g = 112$).

et plus la convergence est rapide et prématurée : le SAG tombe rapidement dans des optima locaux. Penchons nous quelque peu sur cette observation.

Lorsque la population est initialisée, différentes valeurs de Gestalt variables sont générées ou non. Et la probabilité de leurs présences dans cette population initiale décroît en fonction de la taille de la Gestalt variable. Or, lors de la recombinaison, les Gestalt variables sont échangées, de par leur définition, entièrement entre deux individus. Aucune nouvelle valeur qui n'était pas présente dans la population initiale ne peut donc être introduite. Même avec la mutation, seul le complément des valeurs peut être introduit. Ainsi, certaines régions de l'espace de recherche restent-elles inaccessibles. Le SAG visitant très rapidement le potentiel contenu dans la population initiale converge alors vers un optimum local. Nous retrouvons ici l'importance du contexte dans la qualité d'une Gestalt. Lors de la conception a

t **(s)**	H_1 : **meilleur que pour** $\mathcal{G}_1$	p**-valeur (Mann-Whitney U à une queue)**
1	$\mathcal{G}_2$, $\mathcal{G}_3$ et $\mathcal{G}_4$	$< 2{,}2\,10^{-16}$ et $1{,}554\,10^{-3}$ ($\mathcal{G}_4$)
2	$\mathcal{G}_2$ et $\mathcal{G}_3$	$< 2{,}2\,10^{-16}$
3	$\mathcal{G}_2$	$< 2{,}2\,10^{-16}$
4	$\mathcal{G}_2$	$1{,}004\,10^{-4}$
5	$\approx \mathcal{G}_2$	1
6	aucun	> 0.9999

TABLE 5.2 – Le tableau synthétise le résultat des tests statistiques Mann–Whitney U pour comparer les échantillons obtenus par les lentilles $\mathcal{G}_2$, $\mathcal{G}_3$ et $\mathcal{G}_4$ avec l'échantillon de référence obtenu par la lentille $\mathcal{G}_1$. En dernière colonne, il est fourni la p-valeur de l'hypothèse nulle H_0 : la médiane pour la lentille $\mathcal{G}_1$ est plus grande que la médiane de la lentille donnée. Les p-valeurs sont ajustées avec la méthode de Holm pour trois comparaisons. A remarquer que pour $t = 5$ s, la p-valeur affichée est celle obtenue pour l'hypothèse nulle basée sur l'égalité.

priori des lentilles, nous avons implicitement postulé que le contexte était complet, autrement dit que toutes les valeurs possibles des Gestalt variables définies étaient disponibles. Cette assomption implicite n'est pourtant pas réaliste puisque le SAG opère sur une population de taille finie. Ainsi, malgré l'exactitude de la connaissance injectée dans nos diverses lentilles, celles-ci s'avèrent très rapidement inadéquates au contexte.

5.2.6 Conclusion

Au vu des résultats, pouvons-nous à tout le moins conclure que les lentilles ont bel et bien un impact sur le processus de recherche mené par le SAG traitant une instance de CRR. Plus précisément, les observations faites supportent le principe de lentille et de son impact sur le graphe de voisinage. Une lentille conçue en respect de la structure du problème, et ce quelle que soit la manière, se constitue effectivement comme un élément d'assistance positive. Néanmoins, l'adéquation de la lentille est une caractéristique sensible dépendant en particulier de l'état actuel de recherche. Selon ce dernier, la lentille nécessitera d'éventuelles modifications au risque de ne pouvoir assurer son assistance positive ; une conclusion supportant l'aspect dynamique de l'heuristique de la Gestalt telle que nous l'avons définie. Ainsi, une lentille statique pourra au mieux servir de processus d'initialisation efficace.

5.3 Une version biaisée et simplifiée de l'heuristique de la Gestalt

Nous venons de démontrer expérimentalement l'effet indéniable qu'une lentille provoque sur le processus d'un AE tel que le SAG. Nous avons également observé qu'une lentille seule et statique ne pourrait, à terme, que constituer une limitation dans la recherche de l'optimum. Cette observation nous confirme qu'il est nécessaire de faire appel à l'adaptation en ligne des lentilles. Ceci rejoint les discussions conceptuelles menées au Chapitre 3 et justifie un mécanisme tel que celui que nous proposons : l'heuristique de la Gestalt.

Avant d'étudier plus avant la réalisation proposée au Chapitre 4, nous suggérons dans la suite de cette section une preuve expérimentale du principe d'un mécanisme capable d'adapter dynamiquement et en ligne des lentilles afin d'assister positivement un AE dans sa tâche d'optimisation. L'objectif étant une simple preuve de concept, nous nous permettons d'utiliser une version simplifiée et biaisée de l'heuristique de la Gestalt. Aussi, nous élaborerons cette version compte tenu du problème traité et dépouillerons au maximum le mécanisme afin de ne garder que le coeur du concept de la Gestalt : l'élaboration en ligne d'une interprétation de la représentation en termes d'une Gestalt description spécifique au problème.

Dans la continuité de l'expérience précédente, l'AE de base est à nouveau le SAG. Quant au problème, notre choix se porte également sur un problème du même genre que celui du CRR, mais reconnu pour sa plus grande difficulté à être traité par le SAG et sa riche et complexe structure : le problème du SI-et-seulement-Si HIérarchique (SISHI). Au vu de sa définition, une Gestalt variable est dotée d'une relation forte et la représentation des lentilles est spécifiée de telle manière à réduire leur espace de recherche aux seules lentilles pertinentes. La représentation incomplète ainsi choisie permet, comme nous l'expliquerons, un encodage binaire et l'utilisation d'un SAG pour la réalisation de l'heuristique de la Gestalt. Ajoutons que la version est également dénuée de l'opérateur de migration : les sous-populations associées à chaque lentille restent indépendantes et simplement transmises à la nouvelle génération.

L'expérience consistera à comparer l'efficacité du SAG avec et sans assistance en termes d'erreur relative par rapport à l'optimum. Deux types d'assistance seront envisagées : l'une est prodiguée par la réalisation de l'heuristique de la Gestalt que nous venons d'introduire, l'autre par une même réalisation à ceci prêt que le SAG est réduit à une simple génération aléatoire de lentilles. Seront également étudiées l'évolution de la taille des

Gestalt variables et la capacité d'échelle [7] de cette version biaisée de l'heuristique de la Gestalt. Avant d'aborder les résultats à proprement dit, décrivons plus en détails les différents éléments de l'expérience, à commencer par le problème du SISHI.

5.3.1 Problème du SISHI

Partant du CRR, Watson et al. (1998) a proposé et défini une classe de POCs trompeurs, hiérarchiquement décomposables et possédant des blocs constitutifs inséparables. Afin de modéliser cette classe, il en a élaboré une version canonique modélisant toute interdépendance entre blocs constitutifs (de Jong et al., 2004; Forrest and Mitchell, 1993; Watson, 2002) : le problème du SISHI.

Formellement, ce problème se définit sur un vecteur de bits $x = (x_1, \ldots, x_n)$ où $x_i \in \{0, 1\}$. La longueur du vecteur x est égale à $n = k^p$ où k est le nombre de sous-blocs dans un bloc et p est le nombre de niveaux hiérarchiques. Dans la suite, nous considérons toujours $k = 2$. Dès lors, la longueur du vecteur n est déterminée par p. Le SISHI consiste alors à trouver le vecteur x qui maximise la fonction objective récurrente suivante :

$$f_{\text{SISHI}}(B) = \begin{cases} 1 & \text{if } |B| = 1 \\ |B| + f_{\text{SISHI}}(B_L) + f_{\text{SISHI}}(B_R) & \text{if } |B| > 1 \\ & \text{and } \forall i \; b_i = 0 \text{ or } b_i = 1 \\ f_{\text{SISHI}}(B_L) + f_{\text{SISHI}}(B_R) & \text{otherwise,} \end{cases},$$

où B est le bloc de bits $\{b_1, \ldots, b_m\}$, $|B|$ est la taille du bloc B, c'est-à-dire m, b_i est le $i^{\text{ème}}$ élément de B, $B_L = \{b_1, \ldots, b_{m/2}\}$ et $B_R = \{b_{1+m/2}, \ldots, b_m\}$ sont les moitiés gauche et droite du bloc B. Cette fonction f_{SISHI} possède deux maxima globaux obtenus pour les vecteurs ne possédant que des bits à **1** ou **0**.

Afin d'exposer au mieux ce problème, illustrons-le avec un exemple où $k = 2$, $p = 3$ et donc $n = 8$. Si nous considérons la solution candidate $(1,0,1,1,1,0,1,1)$ et que nous résumons f_{SISHI} à f par soucis de clarté, la

7. Par capacité d'échelle, il est entendu la relation entre la taille de l'instance et l'efficacité relative de l'assistance.

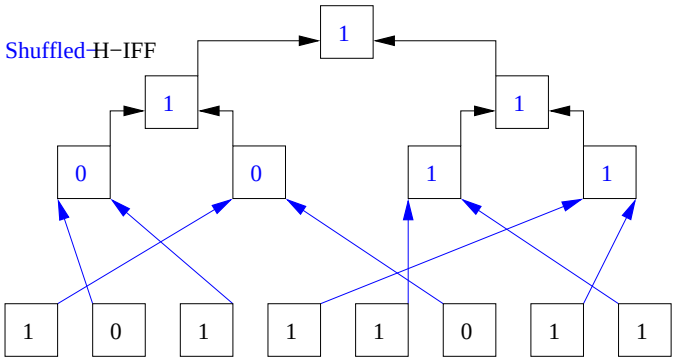


FIGURE 5.4 – La figure illustre le calcul de la valeur objective d'une solution candidate, ici (1,0,1,1,1,0,1,1), selon l'instance du SISHI où $k = 2$ et $p = 3$. Fonction du niveau hiérarchique, un bloc reçoit un bonus particulier s'il est composé exclusivement d'une même valeur de bit. Ici, notre solution candidate est récompensée de deux bonus +2. Considérant la récompense unitaire pour chaque bit, la valeur objective de cette solution est donc de 12.

valeur objective de ce vecteur peut se calculer comme suit :

$$\begin{aligned} f_{\text{SISHI}}(1,0,1,1,1,0,1,1) &= f(1,0,1,1) + f(1,0,1,1) \\ &= 2f(1,0,1,1) \\ &= 2\big(f(1,0) + f(1,1)\big) \\ &= 2\big((f(1) + f(0)) + (2 + f(1) + f(1))\big) \\ &= 2\big((1+1) + (2+1+1)\big) \\ &= 2(2+4) \\ &= 12 \end{aligned}$$

Ce calcul transpose les équivalences logiques hiérarchiques pouvant être opérées sur le vecteur (1,0,1,1,1,0,1,1) comme l'illustre la Fig. 5.4. Ainsi, lorsqu'un bloc se compose exclusivement d'une même valeur de bit, il reçoit un bonus proportionnel au niveau hiérarchique de l'équivalence. De la même manière, nous pouvons calculer les maxima obtenus pour (1,1,1,1,1,1,1,1) et (0,0,0,0,0,0,0,0). Recevant tous les boni possibles, leur valeur $f(1,1,1,1,1,1,1,1) = f(0,0,0,0,0,0,0,0)$ est égale à 32. Et ainsi de suite pour chacune des solutions candidates, nous donnant le paysage d'adaptation présenté à la Fig. 5.5. Ce paysage comporte de nombreux maxima locaux et discontinuités résultant de la définition de f_{SISHI} : à chaque niveau hiérarchique, la distance de Hamming pour atteindre la prochaine meilleure configuration double, empêchant toute progression linéaire (Watson, 2006).

Le SISHI possède un biais par rapport à la relation entre bits : l'ordre de ceux-ci est fixé arbitrairement. Afin d'enlever ce biais, Watson et al.

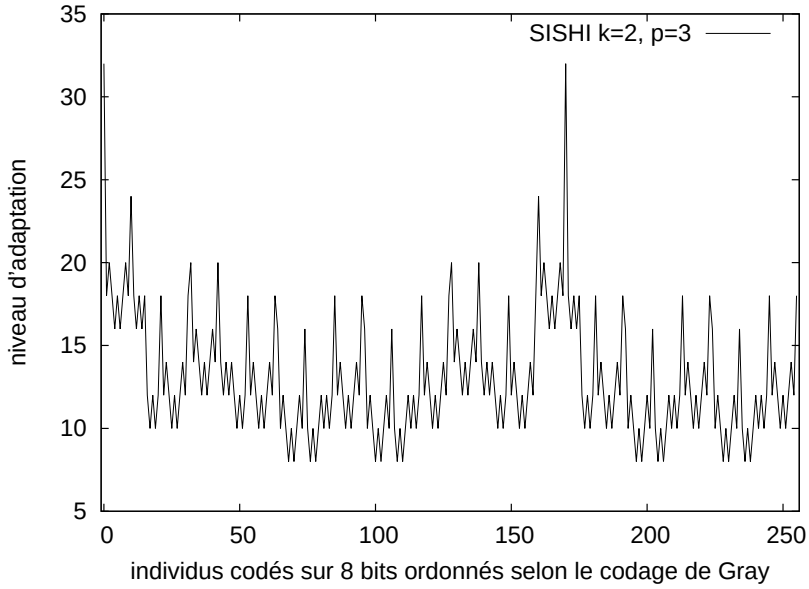


FIGURE 5.5 – Ce graphe représente le paysage d'adaptation de l'instance de SISHI où $k = 2$ et $p = 3$. En abscisse, les solutions candidates sont représentées et classées sous la forme d'un entier obtenu par le codage de Gray. Les optimaux sont en 00000000 (0) et 11111111 (170) et leur valeur est de 32.

(1998) élaborèrent une version du SISHI, appelé le SISHI Mélangé (SISHIM), où la position des bits sont aléatoirement fixée, définissant par là-même une instance particulière du SISHIM. Ce réarrangement implique d'autres valeurs objectives pour les différentes solutions candidates comme illustré par la Fig. 5.6.

L'expérience que nous présentons ci-après est conçue sur la base d'un jeu de cinquante différentes instances de SISHIM où $k = 2$, $p = 7$ et donc $n = 128$ bits. La valeur des maxima atteint 1024. Ces instances sont simplement générées en produisant uniformément une permutation de la position des bits.

5.3.2 SAG et son paramétrage

Afin de traiter les différentes instances de SISHIM, l'AE de base utilisé est à nouveau le SAG[8]. Les paramètres de configuration du SAG sont, rappelons-le, la taille de population N_{pop}, la probabilité de mutation P_μ et la probabilité de croisement P_X. Néanmoins, pour son paramétrage, nous nous bornons cette fois à récupérer la meilleure configuration obtenue par Watson et al. (2002) : $P_X = 1$, $P_\mu = 1/64$ et $N_{pop} = 200$.

8. Pour plus de détails à ce sujet, le lecteur est invité à consulter l'Appendice A.

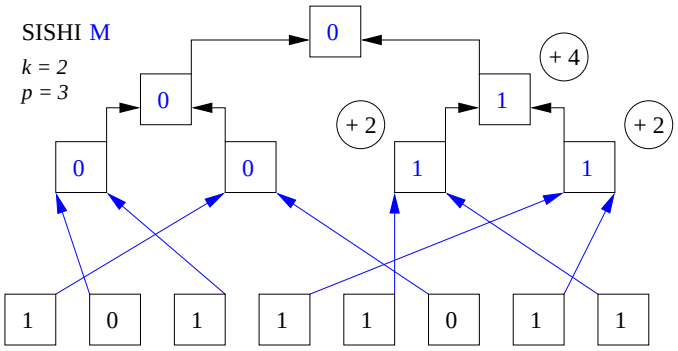


FIGURE 5.6 – La figure illustre le calcul de la valeur objective d'une solution candidate, ici $(1,0,1,1,1,0,1,1)$, selon l'instance du SISHIM où $k = 2$, $p = 3$ et le réarrangement dépeint de la position des bits. Fonction du niveau hiérarchique, un bloc reçoit un bonus particulier s'il est composé exclusivement d'une même valeur de bit. Ici, notre solution candidate est récompensée de deux boni +2 et d'un bonus +4. Considérant la récompense unitaire pour chaque bit, la valeur objective de cette solution est donc de 16.

Nous avons choisi le SAG pour les mêmes raisons que celles ayant justifiée notre choix dans l'expérience précédente. Le SAG est un AG simple et générique dont la non spécialisation justifie une assistance par une heuristique de la Gestalt.

5.3.3 Simplification et spécialisation de l'heuristique de la Gestalt

Pour mener cette expérience, il nous est nécessaire de spécialiser et de simplifier l'implémentation de l'heuristique de la Gestalt. La spécialisation introduite nous permettra de nous assurer qu'il existe bel et bien un moyen de construire des lentilles efficaces. La simplification aura, pour sa part, le rôle de réduire au minimum le mécanisme de l'heuristique de la Gestalt, et de nous assurer ainsi de sa réelle implication dans les résultats obtenus.

Dans la suite, nous décrivons, dans un premier temps, la spécialisation des lentilles au problème du SISHI. Cette spécialisation agit sur deux aspects : sa représentation et son application. Nous terminons cette section en détaillant l'implémentation à proprement dite de l'heuristique.

5.3.3.1 Spécialisation des lentilles au SISHI

Gestalt variable dotée d'une relation inter-variable Le problème du SISHI, nous l'avons vu, récompense les blocs de bits équivalents. Si nous souhaitons spécialiser la Gestalt au SISHI, il convient donc d'enrichir les Gestalt variables d'une relation d'équivalence entre les variables. Cette re-

lation lie dès lors les valeurs des variables, ayant pour conséquence de réduire la valuation d'une Gestalt variable à un sous-ensemble de son domaine, produit des domaines de ses variables constitutives. En résultat de cet enrichissement, l'espace de recherche est projeté sur l'hyperplan défini par les équivalences inter-variables définies implicitement par chacune des Gestalt variables.

Considérons, par exemple, une instance d'un problème binaire encodé sur un vecteur de quatre bits $x = (x_1, x_2, x_3, x_4)$. On définit la Gestalt description $G = g_1, g_2$ avec $g_1 = x_1, x_3$ et $g_2 = x_2, x_4$. Ces Gestalt variables sont donc dotées d'une relation d'équivalence entre leurs variables : toutes deux ne peuvent plus que prendre les valeurs $(1,1)$ et $(0,0)$. Et seules peuvent être représentées les solutions candidates $(0,0,0,0)$, $(0,1,0,1)$, $(1,0,1,0)$ et $(1,1,1,1)$. L'espace de recherche est donc bien projeté sur l'hyperplan défini par le système d'équations

$$\begin{cases} x_1 = x_3 \\ x_2 = x_4 \ . \end{cases}$$

Reste à éclaircir la manière dont la projection est réellement effectuée, et plus précisément, sur quel hyperplan une solution candidate donnée est-elle envoyée. Nous utiliserons la règle de la majorité. Ainsi, une Gestalt variable obtiendra pour valeur, c'est-à-dire celle que prendront toutes ses variables, la valeur prise par la majorité des variables qui la composent, au moment de son application. En cas d'égalité, le choix est simplement tiré au sort. Par exemple, considérant la solution candidate $(1,0,1,1)$ et la Gestalt description $G = g_1$ avec $g_1 = \{x_1, x_2, x_3, x_4\}$, cette solution sera projetée sur $((1,1,1,1))$ puisque la majorité des variables composant g_1 avaient pour valeur 1 avant leur agrégation.

Réduction des possibilités d'agrégation Il est une autre particularité notable propre aux instances du SISHI : la taille des blocs est une puissance de k. Dans notre cas, ce paramètre k est égal à deux. Toujours dans un soucis de spécialisation, il convient dès lors de contraindre la définition des Gestalt variables à n'être constitués que d'une puissance de deux de variables.

Pour terminer, remarquons que la définition du SISHI considère les bits comme étant constitués de bits adjacents. Pour le SISHIM, cette propriété n'est plus respectée. Logiquement, nous ne pouvons donc appliquer cette propriété, telle qu'elle, à la construction des Gestalt variables. Cependant, lors du calcul de la valeur objective, le réarrangement des positions s'effacent au fur et à mesure que l'on monte dans la hiérarchie des blocs. Plus le niveau est haut, plus les blocs sont longs, et plus cette rupture d'ordre est effacée, retrouvant par là-même une propriété d'adjacence. Tout le moins,

cette propriété sera favorisée lors de la construction des Gestalt variables, sans pour autant les contraindre totalement.

Une représentation binaire, biaisée et incomplète des lentilles Afin de rencontrer les contraintes sus-citées, nous suggérons une nouvelle représentation des lentilles. Cette représentation est, comme nous allons le décrire, binaire et incomplète, ce dernier qualificatif nous indiquant que toutes les lentilles, et donc la Gestalt description, possibles ne seront pas toutes représentées. Mais sans plus attendre, décrivons cette représentation.

Considérons une certaine instance de SISHI(M) de taille $n = 2^p$, et une représentation binaire des solutions sous forme de vecteur de bits $x = (x_0, \ldots, x_i, \ldots, x_{n-1})$, où $i = 0, \ldots, n-1$ détermine la position (ou l'index) du bit x_i dans le vecteur. Puisque n est une puissance de deux, l'index i peut être encodé sur p bits, en particulier avec un codage de Gray. Si la lentille est également encodée sur p bits, on peut dès lors effectuer des opérations binaires sur ces deux vecteurs de bits. En effectuant une conjonction entre l'index et la représentation de la lentille, le résultat définit implicitement une partition. Illustrons cette propriété intéressante. Prenons $p = 2$ et donc $n = 4$. L'index i, allant de 0 à 3, est donc codé sur 2 bits selon le codage de Gray : 0 = 00, 1 = 01, 2 = 11 et 3 = 10. Et considérons, par exemple, une lentille codée par 10. La conjonction de cette lentille avec les différents index nous donne respectivement les résultats suivants : 00, 00, 10 et 10. Si l'on interprète ces résultats comme l'appartenance de la variable à cette position à une Gestalt variable, nous avons défini là une Gestalt description à deux Gestalt variables, l'une agrégeant les deux premiers bits, l'autre les deux derniers. Remarquons que l'utilisation de la disjonction nous mènerait à la même interprétation.

Formellement, la représentation des lentilles consiste donc en un vecteur de bits $l = (l_1, \ldots, l_p)$. Ce vecteur l nous permet de reconstruire la Gestalt description associée comme étant l'ensemble G contenant 2^{N_1} Gestalt variables, où N_1 correspond au nombre de bits à la valeur **1** dans le vecteur l. La composition de chacune des Gestalt variables est déterminée par la conjonction de la position i de la variable et du vecteur l : $\{x_i | i \wedge l = e\} = g_e$.

Vérifions si cette représentation rencontre nos objectifs. Comme nous fonctionnons sur la représentation binaire de l'index i et que le vecteur l met à **0** un certain nombre des bits de l'index i, les variables sont réparties selon une puissance de deux dans les différentes 2^{N_1} Gestalt variables. Ces dernières sont donc composées de 2^{p-N_1} variables chacune. Qu'en est-il de l'adjacence ? L'index i est encodé selon le codage de Gray. Ainsi, deux index adjacents sont-ils séparés d'un seul bit. Cette adjacence n'est donc brisée, en ce sens que les bits à ces positions sont répartis dans des Gestalt variables différentes, que lorsque ce bit est conservé, mais que d'autres non, suite

au masquage par l. Ainsi, selon la lentille, l'adjacence est-elle conservée ou non. Nous pouvons donc conclure que cette représentation fait bel et bien écho au problème du SISHI.

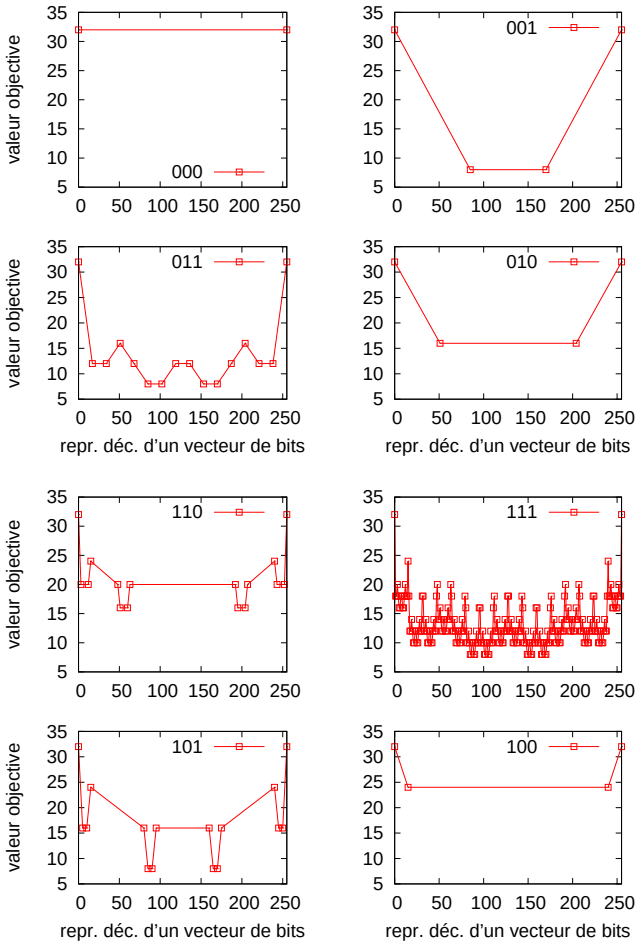


FIGURE 5.7 – La Figure présente la réduction de l'espace de recherche causée par une version de lentille spécialisée pour le SISHI. Il est considéré une instance de SISHI pour $k = 2$ et $p = 3$, dont les solutions candidates sont représentées par un vecteur de huit bits. Les lentilles sont donc représentées par un vecteur de trois bits, et elles sont au nombre de huit : 000, 001, 011, 010, 110, 111, 101 et 100. Dans cet ordre, les paysages d'adaptation résultant de leur application sont présentés, de haut en bas et de gauche à droite. Les solutions candidates sont ordonnées sur l'abscisse selon une représentation décimale classique, et leurs valeurs objectives indiquées selon l'ordonnée. Les solutions encore accessibles après l'application d'une lentille sont indiquées par un carré excepté pour la lentille 111 qui préserve l'espace de recherche dans son entièreté. Rappelons que les solutions optimales sont les vecteurs $(1,1,1,1,1,1,1,1)$ et $(0,0,0,0,0,0,0,0)$, dont la valeur objective est de 32.

Reste à prendre en considération la relation inter-variables qu'induisent désormais les Gestalt variables. Ainsi, lors de l'application d'une lentille, la solution candidate est projetée sur l'hyperplan défini par les équivalences induites par les Gestat variables. De ce fait, l'espace de recherche est réduit à cet hyperplan. Sa cardinalité est égale à $2^{2^{N_1}}$, puisqu'une solution candidate est complètement déterminée par les 2^{N_1} Gestalt variables, chacune ne prenant que deux valeurs possibles, **0** ou **1**. Le nombre de bits à **1** N_1 pouvant varier de 0 à p, la taille de l'hyperplan varie de 2^1 à 2^n, c'est-à-dire l'espace de recherche entier. La projection sur cet hyperplan, lors de l'application d'une lentille, peut être considérée comme un échantillonnage de l'espace de recherche comme la Fig. 5.7 nous l'illustre. Et selon la force de cet échantillonnage, les Gestalt variables définies sont plus ou moins grandes, et en somme, peuvent être associées à un niveau hiérarchique précis de l'instance traitée de SISHI.

En conclusion, les lentilles sont bel et bien spécialisées pour le problème du SISHI par l'intermédiaire de la représentation incomplète et biaisée que nous venons de décrire, mais également de la relation inter-variable que les Gestalt variables se voient désormais dotées. Du fait, de cette nouvelle représentation binaire, l'heuristique de la Gestalt peut être réalisée en conséquence, notamment par un SAG. Cette réalisation constitue le sujet de la section ci-après.

5.3.3.2 Evolution des lentilles

Puisque désormais les lentilles sont représentées au moyen de vecteurs de bits, nous pouvons utiliser un AE capable de travailler sur cette représentation. Le SAG est tout trouvé pour résoudre cette tâche, sa simplicité n'appuyant que d'autant ce choix. Dans la suite, nous exposons les détails de la réalisation simplifiée de l'heuristique de la Gestalt que nous utiliserons dans la présente expérience, en soulignant avec un soin tout particulier les différences avec l'implémentation décrite au Chapitre 4.

Par un SAG Plutôt que d'utiliser un AGG pour évoluer, nous utiliserons donc un SAG [9] auquel nous ajoutons l'heuristique élitiste consistant à conserver le meilleur individu d'une génération à l'autre. Rappelons que les paramètres de configuration du SAG sont la taille de population r, qui fixera également le nombre de sous-populations, les probabilités de recombinaison P_X et de mutation P_μ. Ces paramètres sont simplement fixés à la main aux valeurs suivantes suite à une petite série de tests grossiers : $r = 10$, $P_X = 7/10$ et $P_\mu = 1/128$.

9. Pour plus de détails à ce sujet, le lecteur est invité à consulter l'Appendice A.

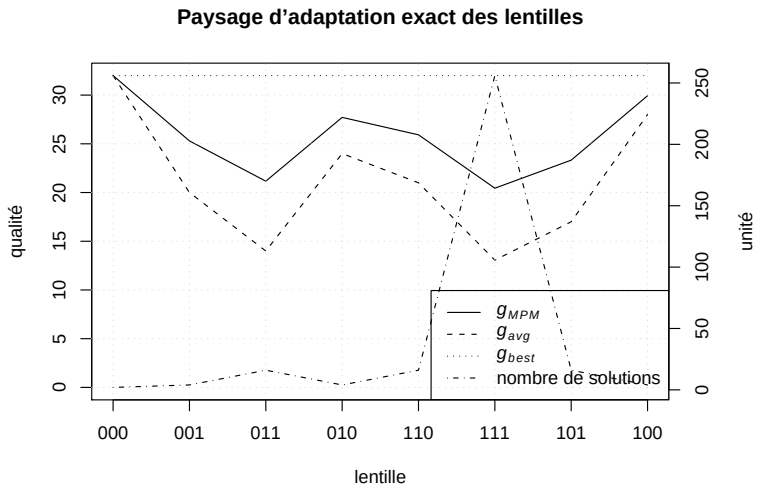


FIGURE 5.8 – La figure présente sur un graphe les paysages d'adaptation exacts des lentilles résultants des fonctions g_{best}, g_{avg} et g_{MPM}. Les lentilles sont classées sur l'abscisse, et l'ordonnée de l'axe gauche indique leur qualité évaluée. L'exactitude du calcul nécessite un nombre plus ou moins grand de solutions candidates, selon la lentille. Ce nombre est indiqué par la courbe portant le même nom selon l'axe de droite.

Par génération aléatoire Afin de s'assurer de l'efficacité du SAG dans la construction des lentilles, les résultats obtenus par son intermédiaire seront comparés à une simple génération aléatoire des lentilles lorsque le critère de leur obsolescence est rempli.

Evaluation Pour les raisons citées à la Sec. 4.3.5, nous choisissons la fonction [10] g_{MPM} pour évaluer les lentilles afin de balancer la force de sélection au sein des sous-populations associées aux lentilles.

Permettons-nous une petite digression à ce propos, en étudiant le profil exact de la qualité des lentilles selon les différentes fonctions d'évaluation. Considérant l'instance de SISHI où $k = 2$ et $p = 3$, nous pouvons calculer exactement la qualité de toutes les lentilles, et dès lors en tracer les paysages d'adaptation, sur la Fig. 5.8, obtenus pour les fonctions d'évaluations g_{best}, g_{avg} et g_{MPM}. Ces paysages possèdent une structure d'une simplicité extrême, contrairement à celui du SISHI, et confortent l'idée que nous défendons au sein de cette thèse. Toutefois, ces évaluations exactes sont à modérer compte tenu du nombre de solutions candidates nécessaires à les calculer. Ainsi, il est très peu probable que la qualité de la lentille 111 obtenue pratiquement soit aussi précise, et ce quelle que soit la fonction

10. Cette fonction se calcule comme étant la racine carré du produit de la qualité du meilleur individu au sein d'une île par la moyenne de la qualité des individus d'une île.

d'évaluation. Il est temps de fermer cette parenthèse et d'aborder le dernier point, la structure de population.

Structure de population Suite à l'application des lentilles, la population des solutions candidates est subdivisée en autant de sous-populations que de lentilles. La population des lentilles étant de taille r et celle des solutions candidates de taille N_{pop}, nous avons r sous-populations de taille $\rho = N_{pop}/r$, dans le cas présent, les sous-populations contiendront 20 individus. Par contre, les sous-populations restent indépendantes, et aucune migration d'individu n'a lieu entre elles. Et lors de la création d'une nouvelle génération de lentille, chacune des sous-populations est transmise au descendant le plus proche de la lentille à laquelle elle était associée. Et la nouvelle génération de lentilles s'opère à un intervalle régulier de générations de solution candidates, à chaque générations $t_m = 1$.

Ceci termine la description de l'implémentation de l'heuristique de la Gestalt, spécialisée au problème du SISHI. Les différences notoires avec la réalisation décrite au Chapitre 4 sont l'utilisation d'une représentation incomplète, binaire et biaisée des lentilles, du SAG pour leur évolution, et de l'indépendance des sous-populations.

5.3.4 Conditions expérimentales

Afin d'étudier expérimentalement l'assistance au SAG prodiguée par l'évolution en ligne des lentilles, l'évolution de la meilleure solution candidate est mesurée en fonction du nombre d'évaluations de f_{SISHI}. L'efficacité du SAG seul est comparée à celles obtenues sous assistance de lentilles générées aléatoirement et évoluées par le SAG. L'efficacité est mesurée sur un ensemble de trente essais de traitement de cinquante instances différentes de SISHIM de 128 bits. Sur ce même ensemble d'instances et d'essais, la taille moyenne des Gestalt variables de la meilleure lentille est également relevée en fonction du nombre d'évaluations de f_{SISHI}.

Afin d'étudier l'effet d'échelle, le SAG assisté de l'heuristique de la Gestalt spécialisée est appliqué à des SISHIM de plusieurs tailles (de 32 à 4096 bits, c'est à dire pour $p \in [5, 12]$ et $k = 2$). Pour chaque taille, cinquante instances différentes sont traitées trente fois. Au cours de chacun de ces essais, il est mesuré le temps processeur réel nécessaire au SAG assisté pour trouver un des deux optima.

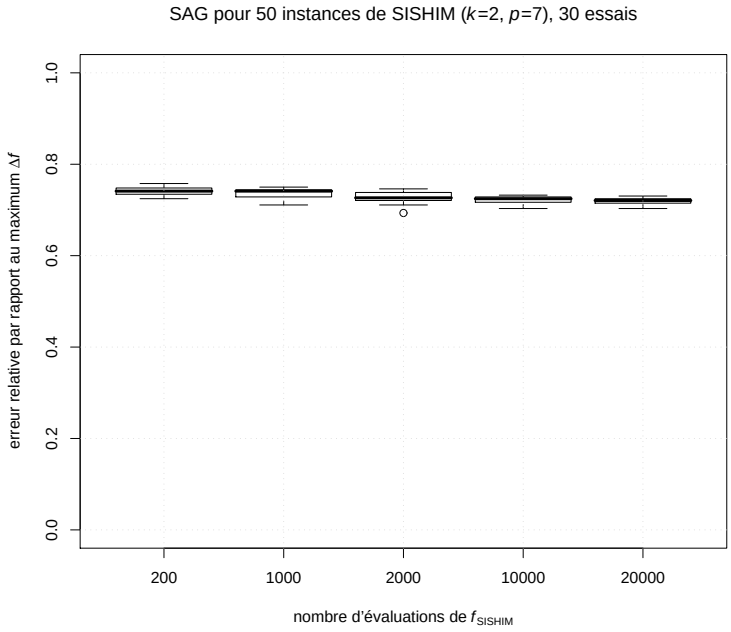


FIGURE 5.9 – Le graphe transpose la mesure d'efficacité du SAG pour traiter cinquante instances de SISHIM de 128 bits ($k = 2$, et $p = 7$), au cours de trente essais. L'évolution de la distribution de l'erreur relative des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du nombre d'évaluations de f_{SISHI}.

5.3.5 Résultats et discussion

La Figure 5.9 présente les résultats obtenus pour le SAG seul. Ces résultats expriment l'évolution de la distribution de l'erreur relative Δf des meilleurs solutions candidates obtenues au cours des trente essais de traitement de chacune des cinquante instances de SISHIM. La distribution est représentée sous la forme d'une boîte à moustaches. Nous pouvons noter que la progression du SAG est extrêmement lente et converge sans pouvoir descendre en dessous des 7/10 d'erreur relative [11], observation rejoignant celles faites par Watson (2002). Traiter une instance de SISHIM est une tâche ardue pour le SAG.

La Figure 5.10 présente, pour sa part, les résultats obtenus pour le SAG assisté par des lentilles générées aléatoirement. Nous utilisons les mêmes modalités de représentation que précédemment. Est également présentée l'évolution de la qualité de la meilleure lentille. Cette fois, de temps à autre, le SAG trouve l'optimum, et ce grâce à des lentilles adéquates obtenues par chance. Et plus les générations se succèdent, plus la probabilité de générer une telle lentille augmente. Ce qui permet d'expliquer la légère

11. Ce qui équivaut à valeur objective d'un peu plus de 300.

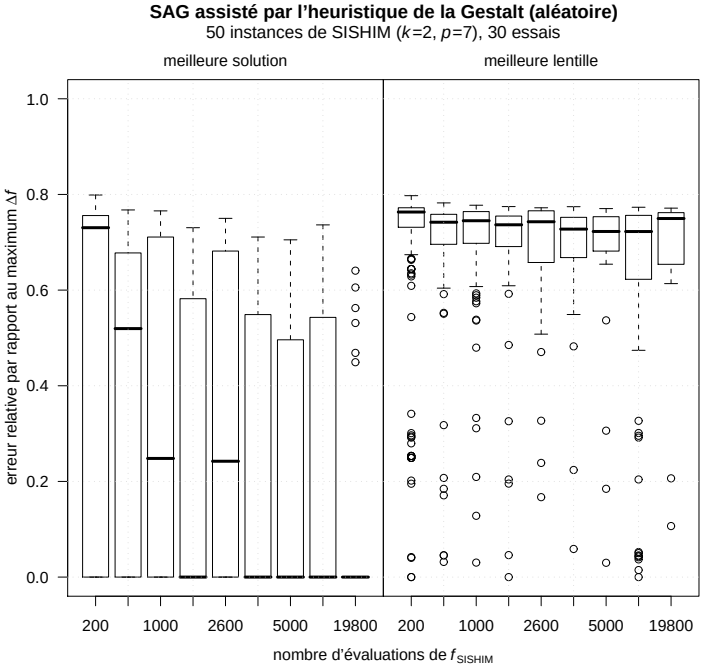


FIGURE 5.10 – Les graphes transposent la mesure d'efficacité de l'assistance au SAG prodiguée par des lentilles générées aléatoirement, le SAG traitant trente fois les cinquante différentes instances de SISHIM de 128 bits ($k = 2$, et $p = 7$). Sur le graphe de gauche, l'évolution de la distribution de l'erreur relative des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du nombre d'évaluations de f_{SISHI}. Celui de droite, quant à lui, expose l'évolution de la distribution de l'erreur relative des meilleures lentilles.

tendance positive au niveau des meilleures solutions candidates trouvées. Ces résultats sont importants car ils vont nous servir de référence pour évaluer la réalisation de l'heuristique de la Gestalt au moyen d'un SAG.

Finalement, la Figure 5.11 présente les résultats obtenus pour le SAG assisté par des lentilles évoluées au moyen d'un SAG. A nouveau, les mêmes modalités de présentation sont utilisées. Il ne faut attendre que la deuxième génération de lentille pour que l'un des deux maxima soit trouvé à tous les coups. Et lorsque que l'on compare l'évolution de la distribution de la meilleure lentille avec celle obtenue pour des lentilles générées aléatoirement, il est particulièrement intéressant de noter la différence d'évolution : les lentilles aléatoires ne démontrent aucune tendance alors que, dans le cas présent, une réelle tendance positive peut être relevée. Dès à présent, nous pouvons à tout le moins conclure que l'évolution des lentilles démontre sa réelle participation dans le processus d'assistance

Histoire de mettre en perspective les résultats précédents, nous nous penchons sur l'évolution de la taille des Gestalt variables lorsque les lentilles

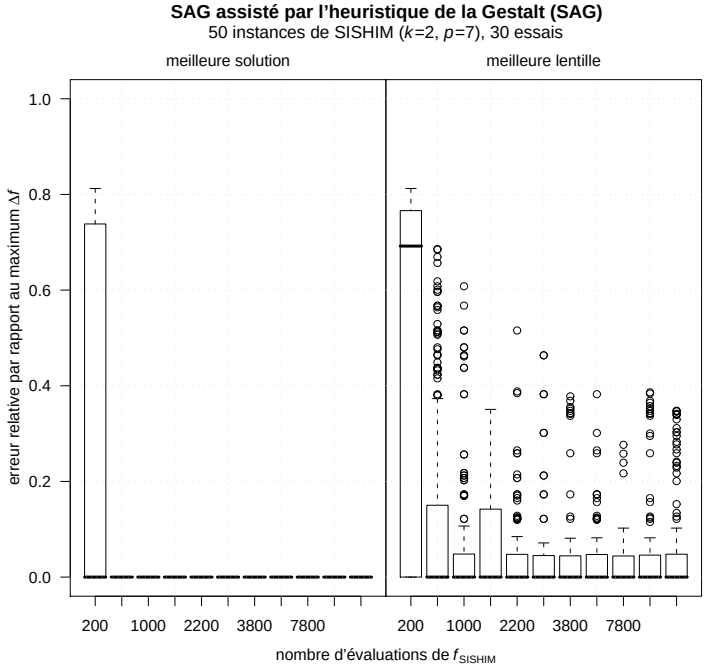


FIGURE 5.11 – Les graphes transposent la mesure d'efficacité de l'assistance au SAG prodiguée par des lentilles évoluées au moyen d'un autre SAG. Les instances, traitées trente fois chacune et du type de SISHIM de 128 bits ($k = 2$, et $p = 7$), sont au nombre de cinquante. Sur le graphe de gauche, l'évolution de la distribution de l'erreur relative des meilleurs solutions candidates est représentée sous forme de boîtes à moustaches en fonction du nombre d'évaluations de f_{SISHI}. Celui de droite, quant à lui, expose l'évolution de la distribution de l'erreur relative des meilleures lentilles.

sont générées aléatoirement ou évoluées au moyen d'un SAG. Tel est l'objet de la Fig. 5.12. Celle-ci présente l'évolution de la taille moyenne (et de son écart type) des Gestalt variables induites par la meilleure lentille en fonction du nombre d'évaluations de la fonction f_{SISHI}. Comme nous pouvons l'observer, il suffit d'une génération pour que le SAG fournisse des lentilles induisant des Gestalt variables de grande, voire très grande taille. Ceci n'est que le reflet de la spécialisation de l'heuristique de la Gestalt au problème du SISHI. Très rapidement, les lentilles sélectionnées sont celles qui reformulent le problème à un niveau hiérarchique élevé, agrégeant par là-même un très grand nombres de variables. Les lentilles générées aléatoirement, elles, se cantonnent à des tailles beaucoup plus petites de Gestalt variables. Ce dernier comportement peut s'expliquer par le fait que les lentilles sont générées aléatoirement grâce à une distribution uniforme. Or, elles sont encodées par un vecteur de bits, et par conséquent le nombre de bits à 1 est en moyenne de 3,5, c'est-à-dire la moitié de la longueur du vecteur. Ceci

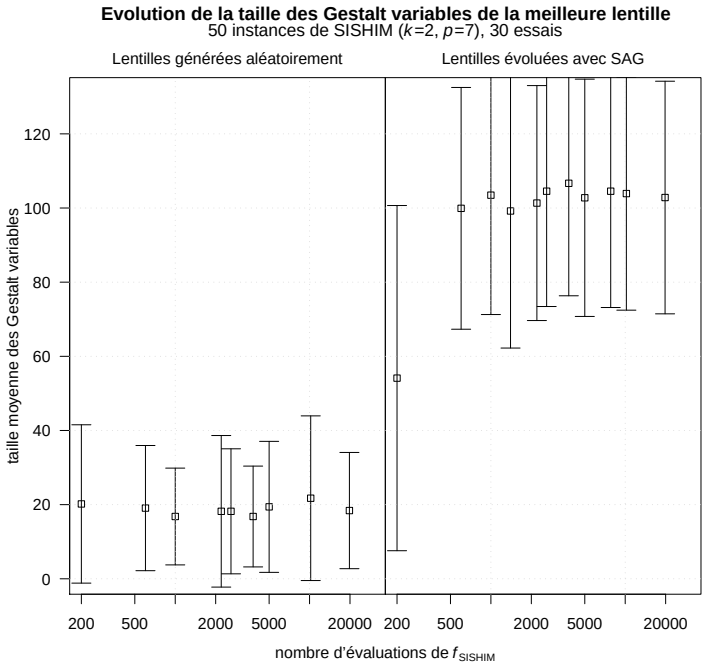


FIGURE 5.12 – La figure compare l'évolution de la taille moyenne, et de son écart type, des Gestalt variables induites par des lentilles générées aléatoirement (sur le graphe de gauche) ou des lentilles évoluées au moyen d'un SAG (sur le graphe de droite), fonction du nombre d'évaluations de f_{SISHI}.

nous donne une taille moyenne $2^{7-3,5} \approx 11$, taille correspondant à l'ordre de grandeur des résultats expérimentaux.

Nous terminons l'exposition des résultats avec l'étude des capacités d'échelle du SAG une fois assisté par l'heuristique de la Gestalt spécialisée. Il s'agit ici d'étudier l'impact de la taille de l'instance de SISHIM sur le temps processeur nécessaire à trouver l'un des deux optima. Rappelons que pour chaque taille, allant de 32 à 4096 bits, cinquante instances différentes sont traitées chacune trente fois. La Figure 5.13 présente les résultats de cette étude. Chaque fois que la taille est doublée, le temps nécessaire est à peu près doublé également. Nous avons donc une progression linéaire du temps processeur en fonction de la taille de l'instance. Ce comportement résulte clairement de la spécialisation des lentilles, spécialisation pouvant conduire à une certaine réduction de l'espace de recherche des solutions, biaisant ainsi positivement la recherche qui y est menée. En effet, selon la lentille, l'espace de recherche est projeté sur un hyperplan, autrement dit un de ses sous-ensembles, de cardinalité égale à $2^{2^{N_1}}$. La lentille peut donc devenir un puissant moyen de réduire la complexité du problème, conséquence fort normale compte tenu de notre volonté de spécialiser notre heuristique au problème du SISHIM.

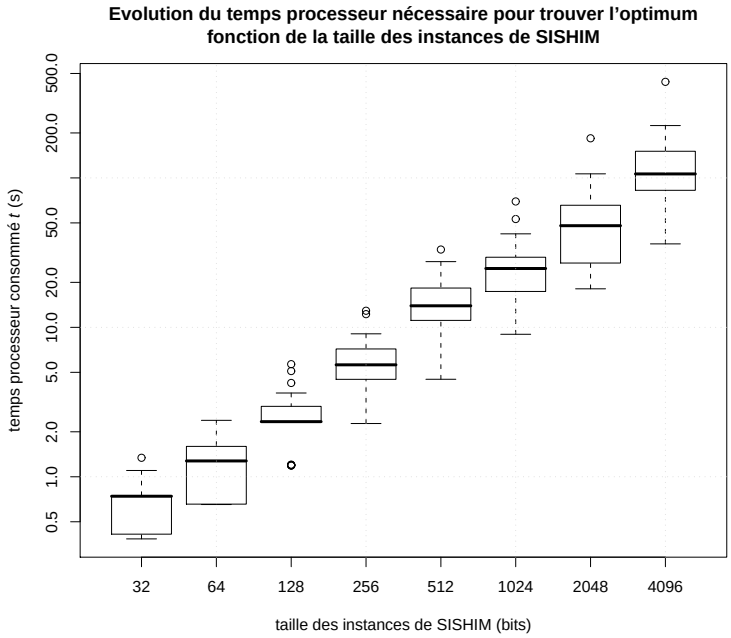


FIGURE 5.13 – Le graphe résume l'évolution du temps processeur requis par le SAG assisté par l'heuristique de la Gestalt spécialisée pour trouver un des deux optima en fonction de la taille de l'instance de SISHIM. Pour chaque taille, cinquante différentes instances sont traitées, chacune, trente fois.

5.3.6 Conclusion

L'objectif de l'expérience que nous venons de présenter était d'éprouver la possibilité d'élaborer un mécanisme capable de modifier les lentilles au cours de l'assistance. Lors de la première expérience, nous avons souligné l'importance du contexte, autrement dit l'ensemble des solutions candidates trouvées. Ajoutons que l'espace de recherche des lentilles est immensément grand. Ces contraintes rendant l'épreuve d'autant plus difficile, nous nous sommes assurés de les réduire au maximum par la simplification et la spécialisation de l'heuristique de la Gestalt. Et de cette manière, nous pouvons nous concentrer sur la seule question posée.

Les divers résultats ont démontré l'intérêt d'un mécanisme adaptant les lentilles en cours d'optimisation. En effet, le SAG dédié aux lentilles réussit à concevoir des lentilles capables d'une réelle assistance positive au processus d'optimisation. Non seulement le gain par rapport aux résultats obtenus sans assistance est indéniable, mais il est bien meilleur à celui obtenu sous l'assistance de lentilles générées aléatoirement.

Reste à savoir si cette démonstration expérimentale positive n'est pas le seul produit de la spécialisation effectuée. D'une part, ce succès nous mon-

tre que les lentilles s'avèrent être un moyen d'inoculer de la connaissance a priori sur un problème par la simple reformulation de sa représentation. Les lentilles jouent bel et bien un rôle dans le processus de recherche. L'assistance est bien réelle. Et d'autre part, la spécialisation n'a pour conséquence que la restriction de l'espace de recherche des lentilles et l'amélioration de leur contexte. Il ne s'agit donc pas d'un tout autre cas de figure, mais d'un sous-cas. A tout le moins, nous pouvons conclure qu'il est possible d'élaborer des lentilles efficaces, lorsque le contexte le permet.

La spécialisation n'est donc qu'une paire de béquilles dont il nous faut nous passer s'il l'on souhaite concevoir une heuristique de la Gestalt générique. Reste à savoir si la réalisation que nous proposons le permet. Une question que nous posons dans le cadre de l'expérience présentée dans la prochaine section.

5.4 Etude de l'implémentation de l'heuristique de la Gestalt

Au cours de l'expérience précédente, nous avons montré qu'il était conceptuellement possible d'élaborer un mécanisme capable d'assister dynamiquement et en ligne le processus de recherche mené par une métaheuristique telle un AG. Ainsi, non seulement les lentilles s'avèrent un réel moyen d'assistance de ce processus, mais il est possible de produire automatiquement des lentilles efficaces. Toutefois, cette preuve de concept, qui s'appuie sur une version simplifiée et spécialisée de l'heuristique, ne nous permet nullement de valider la réalisation proposée et discutée au Chapitre 4. Tel est l'objectif de l'expérience qui fait le sujet de cette section-ci.

Pour ce faire, il nous faut avant tout choisir une AE et un ou plusieurs problèmes. Afin d'assurer une continuité avec les discussions expérimentales précédentes, l'AE est à nouveau le SAG. Nous lui soumettons deux POCs différents, formant deux extrêmes de complexité : le PMU possédant une simple structure et le SISHIM 2 une structure riche et complexe. Ce dernier POC devrait offrir l'occasion à notre heuristique de la Gestalt de démontrer ses qualités, puisqu'elle vise avant tout de découvrir et exploiter une structure au profit d'un processus d'optimisation.

L'expérience consistera alors à comparer l'efficacité du SAG avec et sans assistance en termes d'erreur relative par rapport à l'optimum de telle sorte à souligner ou non l'assistance positive de l'heuristique de la Gestalt. Reste à savoir si son éventuel succès peut réellement lui être attribué. L'implémentation de l'heuristique met effectivement en jeu un certain nombre de mécanismes à des fins différentes : contextualisation, investigation ou as-

sistance. Pour exemple, nous pouvons citer le modèle de multi-populations utilisé principalement dans la contextualisation des lentilles. Il serait possible que son intervention dépasse ce seul cadre, faut-il donc dès lors nous en assurer. C'est pourquoi nous envisagerons trois formes d'assistance du SAG, dont deux formes constitueront des versions de l'heuristique de la Gestalt dénuées de l'un ou l'autre de ses composants. La comparaison des assistances ainsi prodiguées nous permettra de justifier le réel rôle de l'heuristique de la Gestalt dans les résultats obtenus.

A cette même fin, nous étudierons également l'évolution de la taille des Gestalt variables et la capacité d'échelle [12] des diverses versions testées de l'heuristique de la Gestalt. Ayant terminé d'esquisser le programme prévu par cette expérience, nous pouvons exposer plus avant les différents éléments pris en compte.

5.4.1 Problèmes traités : PMU et SISHIM 2

Afin d'analyser le rôle des différents éléments de l'implémentation de l'heuristique de la Gestalt, il nous est nécessaire de choisir des POCs permettant de souligner l'intérêt de l'heuristique de la Gestalt. Cette dernière, rappelons-le, base son mécanisme fondamental sur la reformulation de la représentation du problème traité de telle manière à exploiter implicitement certaines de ses propriétés. Il convient dès lors de choisir deux types de POC :

- un POC simple avec une structure pauvre et linéaire, et
- un POC trompeur avec une structure riche et complexe.

Compte tenu de l'heuristique de la Gestalt, nous devrions nous attendre à une meilleure assistance sur le second type. Cette motivation nous guidant, nous portons notre choix sur le PMU et le problème du SISHIM 2.

5.4.1.1 PMU

Le PMU, exposé au Chapitre 2, à la Sec. 2.1.2.2, est un simple problème de comptage de **1** dans une chaîne de bits. Ainsi, considérant un vecteur de n bits $x = (x_1, \ldots, x_n)$, le PMU consiste à trouver le maximum de la fonction :

$$f_{\text{PMU}}(x) = \sum_{i=1}^{n} x_i \ .$$

La valeur du maximum x^* vaut simplement la taille de l'instance considérée, c'est-à-dire $f(x*) = n$.

12. Par capacité d'échelle, il est entendu la relation entre la taille de l'instance et l'efficacité relative de l'assistance.

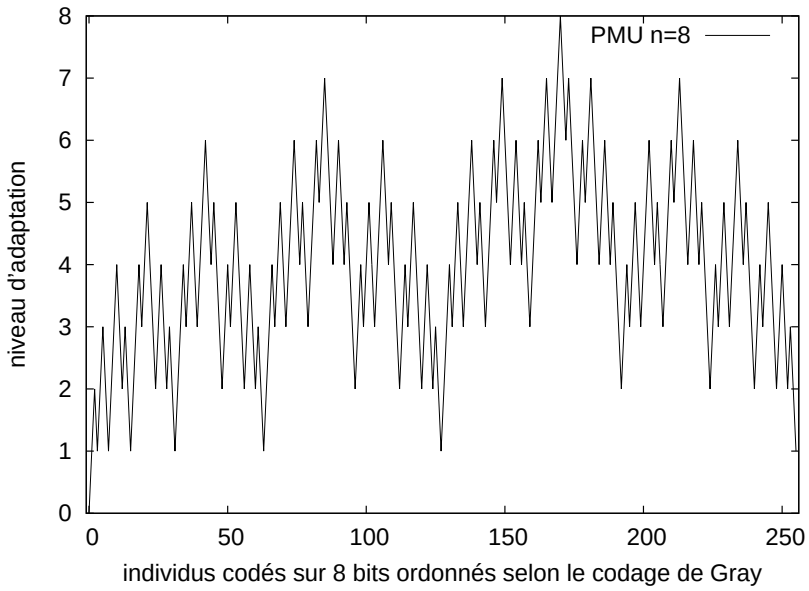


FIGURE 5.14 – Ce graphe représente le paysage d'adaptation de l'instance de PMU de 8 bits. En abscisse, les solutions candidates sont représentées et classées sous la forme d'un entier obtenu par le codage de Gray. L'optimum est en 11111111 (170) et sa valeur est de 8.

La Figure 5.14 présente le paysage d'adaptation d'une instance de PMU de 8 bits. Bien qu'ayant plusieurs optima locaux, le paysage présente une certaine continuité graduelle entre les voisins.

Lors de l'expérience présentée ci-dessous, seront considérées cinq tailles d'instances formant une progression géométrique de raison deux : 1000, 2000, 4000, 8000 et 16000.

5.4.1.2 SISHIM 2

Le problème du SISHIM, exposé à la Sec. 5.3.1, possède un biais en termes de complémentarité des ses deux optima. Afin de retirer ce biais, Watson (2001) élabora une nouvelle version du SISHIM, appelé le SISHIM 2, où les deux optima sont fixés arbitrairement, définissant par là-même une instance particulière du SISHIM 2.

Formellement, le problème du SISHIM 2 consiste en un problème du SISHIM dont la fonction objective à maximiser est désormais :

$$f_{\text{SISHIM2}}(x) = f_{\text{SISHIM}}(x \oplus g_1) + f_{\text{SISHIM}}(x \oplus g_2) \ ,$$

où le symbole $\oplus$ désigne la disjonction exclusive, et g_1 and g_2 sont les deux optima arbitraires, deux vecteurs de même taille que celle de x. En

pratique, une instance de SISHIM 2 est obtenue après génération aléatoire d'une permutation de la position des bits et de la génération aléatoire des deux optima g_1 et g_2.

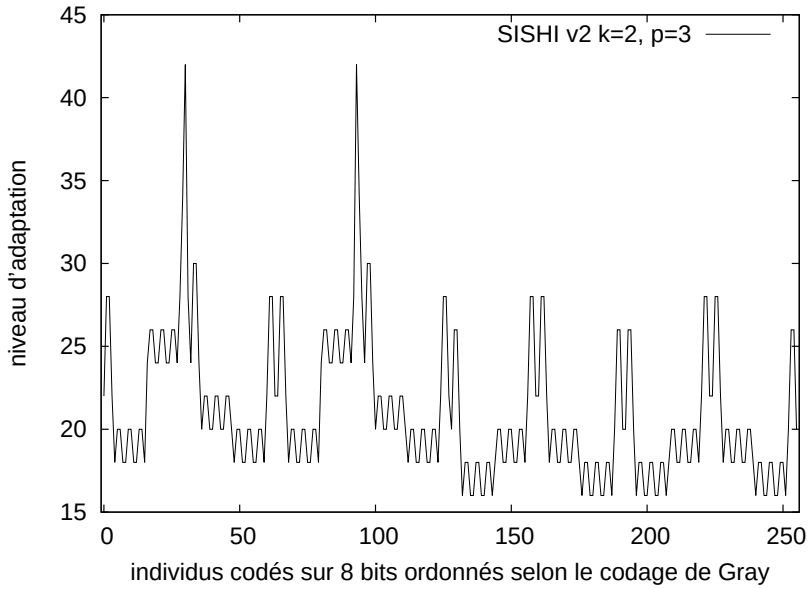


FIGURE 5.15 – Ce graphe représente le paysage d'adaptation de l'instance de SISHI 2 où $k = 2$ et $p = 3$. En abscisse, les solutions candidates sont représentées et classées sous la forme d'un entier obtenu par le codage de Gray. Les optimaux sont en 01110011 (30) et 00010001 (93) et leur valeur est de 42.

La Figure 5.15 nous permet de visualiser clairement ce qui différencie la deuxième version du SISHI de sa version originale. Cette fois-ci les optimaux sont en 01110011 (30) et 00010001 (93) et du fait de leur proximité leur valeur est de 42 et non plus 32.

L'expérience que nous exposons ci-après met en jeu cent différentes instances de SISHIM 2 où $k = 2$ et p prend successivement les valeurs de huit à treize, nous donnant ainsi les six tailles formant une progression géométrique de raison deux : 256, 512, 1024, 2048, 4096 et 8192.

5.4.2 SAG et son paramétrage

Les problèmes que nous venons d'exposer se représentent naturellement au moyen de variables binaires, tout comme les problèmes abordés dans les expériences précédentes. Aussi, utilisons-nous à nouveau un SAG [13]. Pour rappel, ses paramètres de configuration sont la taille de population N_{pop}, la probabilité de mutation P_μ et la probabilité de croisement P_X. Nous le choi-

13. Pour plus de détails à ce sujet, le lecteur est invité à consulter l'Appendice A.

Paramètre	Valeurs possibles	Meilleure valeur
N_{pop}	$\{10, 50, 100, 250, 500, 750, 1000\}$	100
P_μ	$\{0, 1/10000, 1/1000, 1/100, 1/10\}$	1/1000
P_X	$\{0;0,1;0,2;0,3;0,4;0,5;0,6;0,7;0,8;0,9;1\}$	0,5

TABLE 5.3 – Le tableau présente, pour le PMU, les différents ensembles de valeurs pour les paramètres du SAG que sont la taille de population N_{pop}, la probabilité de mutation P_μ et la probabilité de recombinaison P_X. Leur arrangement forme une configuration du SAG afin de traiter l'instance de PMU de 1000 bits. Chaque configuration se voit accorder soixante secondes de temps processeur.

sissons pour les mêmes raisons qui nous y ont poussés lors des expériences exposées en début de ce chapitre : l'heuristique de Gestalt n'a réellement de sens que dans l'assistance d'une métaheuristique générique. Ajoutons-y que notre critère du choix des problèmes restent pertinent puisque le SISHIM 2 est réputé bien plus difficile pour le SAG que ne l'est le PMU.

Pour chaque problème, le SAG est paramétré automatiquement à l'aide de l'algorithme F-Race, présenté en début de ce chapitre, sur les instances de la plus petite taille. La meilleure configuration trouvée sera ensuite utilisée invariablement, quelle que soit l'instance traitée, et quel que soit l'ajout d'algorithme qui lui sera fait. De cette manière, nous nous mettons dans un cas de figure où l'on espère de l'heuristique de la Gestalt des capacités d'adaptation. Mais nous reviendrons sur cet aspect méthodologique un peu plus loin, lors de la description expérimentale à proprement dit.

Pour le PMU, chaque configuration est testée sur une instance de 1000 bits, pour un temps processeur accordé de soixante secondes. Au total, trois cent quatre-vingt-cinq configurations différentes sont testées et produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.3. La meilleure configuration obtenue est une population de 100 individus, une probabilité de mutation de 1/1000, et une probabilité de recombinaison de 0,5.

Pour le SISHIM 2, chaque configuration est testée sur un jeu de cent instances de 256 bits générées aléatoirement, pour un temps processeur accordé de cent vingt secondes. Au total, mille quatre cents configurations différentes sont testées et produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.4. La meilleure configuration obtenue est une population de 250 individus, une probabilité de mutation de 1/1024, et une probabilité de recombinaison de 0,15. Remarquons que la probabilité de recombinaison est très faible. Pour le CRR, nous avions obtenu également un faible taux de recombinaison (voir la Sec. 5.2.2). Ains, ce paramétrage implique un rôle important de la mutation. D'une certaine manière, ce faible taux de recombinaison empêche

Paramètre	Valeurs possibles	Meilleure valeur
N_{pop}	{100,200,250,300,350,400,500,750,1000, 2000}	250
P_μ	{0,1/4096,1/2048,1/1024,1/512,1/256, 1/128,1/64,1/32,1/16}	1/1024
P_X	{0;0,05;0,1;0,15;0,2;0,25;0,3;0,4;0,5; 0,6;0,7;0,8;0,9;1}	0,15

TABLE 5.4 – Le tableau présente, pour le SISHIM 2, les différents ensembles de valeurs pour les paramètres du SAG que sont la taille de population N_{pop}, la probabilité de mutation P_μ et la probabilité de recombinaison P_X. Leur arrangement forme une configuration du SAG afin de traiter des instances de SISHIM 2 de 256 bits. Chaque configuration est testée sur un jeu de cent des ces instances générées aléatoirement et se voit accorder cent vingt secondes de temps processeur.

une convergence prématurée, et en somme, F-Race a trouvé un équilibre entre intensification et diversification approprié à ce type de problème.

5.4.3 L'heuristique de la Gestalt et son paramétrage

L'expérience a pour principal objectif d'étudier les différents éléments de l'implémentation de l'heuristique de la Gestalt que nous avons exposée à la Sec. 4.3 du Chapitre 4. Cette implémentation consiste en un AGG évoluant une population de r lentilles sans recouvrement au moyen d'opérateurs génétiques de groupement (paramétrés par les probabilités de croisement P_X, d'inversion P_I et des trois mutations $P_{R\mu}$, $P_{S\mu}$ et $P_{M\mu}$), d'une sélection EUS, d'un élitisme [14] et d'une initialisation uniforme dans la taille et le nombre des Gestalt variables. Chaque lentille est appliquée à une sous-population distincte de ρ solutions candidates, lui permettant en particulier d'évaluer sa qualité selon une certain fonction d'évaluation g. Cette subdivision de la population en r îles donne lieu à une éventuelle migration entre elles, dont la fréquence t_m et la taille ρ_m peuvent être fixées. Les émigrants sont sélectionnés par EUS et remplacent des individus aléatoires dans leur nouvelle population.

Dans cette implémentation, nous pouvons distinguer trois composants fondamentaux :

1. la structure en sous-populations et la dynamique migratoire,
2. l'application de lentilles à ces sous-populations, et
3. l'évolution des lentilles sur base de l'évaluation des sous-populations.

Ces composants s'additionnent dans l'ordre pour former l'implémentation complète en y assurant différents rôles. Afin d'analyser leurs rôles dans

14. La meilleure lentille et sa sous-population sont copiées dans la nouvelle génération.

Paramètre	Valeurs possibles	Meilleure valeur
r	$\{1,2,5,10\}$	5
ρ	N_{pop}/r	20
ρ_m	$\{1/10,1/5,1/2\}$	1/10
t_m	$\{2,5,10\}$	5
P_X	$\{1/4,1/2,3/4,1\}$	1
P_I	$\{1/4,1/2,3/4,1\}$	3/4
$P_{R\mu}$	$\{1/10000,1/1000,1/100\}$	1/1000
$P_{S\mu}$	$\{1/4,1/2,3/4,1\}$	1/4
$P_{M\mu}$	$\{1/4,1/2,3/4,1\}$	1/2
g	$\{g_{best}, g_{avg}, g_{MPM}\}$	g_{best}

TABLE 5.5 – Le tableau présente les différents ensembles de valeurs pour les paramètres de l'heuristique de la Gestalt assistant le SAG de paramètres $N_{pop} = 100$, $P_\mu = 1/1000$ et $P_X = 0,5$: la taille de la population de lentilles ou le nombre de sous-populations r, la talle des sous-populations de solutions candidates ρ, la taille de migration ρ_m comme fraction de ρ, la fréquence de migration t_m, les probabilités des opérateurs génétiques de groupement (la recombinaison P_X, l'inversion P_I et trois mutations $P_{R\mu}$, $P_{S\mu}$ et $P_{M\mu}$) et la fonction d'évaluation des lentilles g. Leur arrangement forme une configuration de l'heuristique de la Gestalt afin d'assister le SAG à traiter l'instance de PMU de 1000 bits. Chaque configuration se voit accorder dix secondes de temps processeur.

cet ensemble, nous distinguerons deux formes partielles. La première consiste simplement en l'application du modèle multi-populations au SAG. La deuxième, comprenant la première, ajoute l'application de lentilles générées aléatoirement à chacune des sous-populations de solutions candidates. Partant de cette deuxième forme partielle, l'implémentation complète ajoute le mécanisme d'évolution des lentilles. La comparaison des résultats obtenus au moyen de ces diverses formes nous permettra dès lors de justifier ou non l'implémentation suggérée de l'heuristique de la Gestalt.

Pour chaque problème, l'heuristique de la Gestalt, dans sa forme complète, est paramétrée automatiquement à l'aide de l'algorithme F-Race, présenté en début de ce chapitre, afin d'assister le SAG dans sa meilleure configuration (voir les Tableaux 5.3 et 5.4) à traiter les instances de la plus petite taille. La meilleure configuration trouvée sera ensuite utilisée invariablement, quelle que soit l'instance traitée et quelle que soit la forme considérée de son implémentation. Nous reviendrons sur cet aspect méthodologique à la prochaine section.

Pour le PMU, chaque configuration est testée sur une instance de 1000 bits, pour un temps processeur accordé de dix secondes. Au total, quatre-vingt-deux mille cinq cent nonante-deux configurations différentes sont testées et produites selon différentes possibilités de valeur pour chaque

Paramètre	Valeurs possibles	Meilleure valeur
r	$\{20, 25, 30\}$	25
ρ	N_{pop}/r	10
t_m	$\{1, 2, 3, 5\}$	2
ρ_m	$\{3/20, 1/5, 1/4\}$	1/5
P_X	$\{1/4, 1/2, 3/4, 1\}$	1
P_I	$\{1/4, 1/2, 3/4, 1\}$	1/2
$P_{R\mu}$	$\{1/256, 1/128, 1/64\}$	1/128
$P_{S\mu}$	$\{1/4, 1/2, 3/4, 1\}$	1
$P_{M\mu}$	$\{1/4, 1/2, 3/4, 1\}$	1/4
g	$\{g_{best}, g_{avg}, g_{MPM}\}$	g_{best}

TABLE 5.6 – Le tableau présente les différents ensembles de valeurs pour les paramètres de l'heuristique de la Gestalt assistant le SAG de paramètres $N_{pop} = 250$, $P_\mu = 1/1024$ et $P_X = 0,15$: la taille de la population de lentilles ou le nombre de sous-populations r, la talle des sous-populations de solutions candidates ρ, la taille de migration ρ_m comme fraction de ρ, la fréquence de migration t_m, les probabilités des opérateurs génétiques de groupement (la recombinaison P_X, l'inversion P_I et trois mutations $P_{R\mu}$, $P_{S\mu}$ et $P_{M\mu}$) et la fonction d'évaluation des lentilles g. Leur arrangement forme une configuration de l'heuristique de la Gestalt afin d'assister le SAG à traiter des instances de SISHIM 2 de 256 bits. Chaque configuration est testée sur un jeu de cent de ces instances générées aléatoirement et se voit accorder dix secondes de temps processeur.

paramètre, telles que répertoriées dans le Tableau 5.5, tableau reprenant la meilleure configuration.

Pour le SISHIM 2, chaque configuration est testée sur un jeu de cent instances de 256 bits, pour un temps processeur accordé de dix secondes. Au total, quatre-vingt-deux mille cinq cent nonante-deux configurations différentes sont testées et produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.5, tableau reprenant la meilleure configuration.

5.4.4 Conditions expérimentales

L'objectif de l'expérience est de valider l'implémentation de l'heuristique de la Gestalt présentée à la Sec. 4.3. Pour ce faire, nous prenons pour référence l'efficacité du SAG à traiter nos deux problèmes, le PMU et le SISHIM 2. Cette efficacité est comparée à celles obtenues avec le SAG assisté par trois modules algorithmiques, comme discutées à la section précédente :

1. le modèle de multi-populations, que nous noterons *Multi-Pop*,
2. la génération aléatoire et l'application de lentilles au modèle de multi-populations, que nous noterons *Gestalt Aléatoire*, et

3. l'évolution par un AGG et l'application de lentilles au modèle de multi-populations, que nous noterons *Gestalt AGG*.

Toutes choses étant égales par ailleurs, l'implémentation *Gestalt AGG* sera donc validée si et seulement si l'efficacité obtenue sous son assistance est meilleur que le SAG seul et sous assistance des compléments *Multi-Pop* et *Gestalt Aléatoire*.

La configuration du SAG, quelle que soit l'assistance prodiguée, correspond à la meilleure configuration du SAG obtenue sur les instances de la plus petite taille, c'est-à-dire 1000 bits pour le PMU et 256 bits pour le SISHIM 2. La configuration des modules d'assistance correspond à la meilleure configuration du module *Gestalt AGG* assistant le SAG dans les mêmes conditions. Ceci nous permet, d'une part de tester les capacités d'adaptation des algorithmes lors de l'augmentation de la taille des instances, et d'autre part d'étudier dans les mêmes conditions les différentes parties algorithmiques qui composent *Gestalt AGG*.

L'efficacité est mesurée comme l'évolution de l'erreur relative par rapport à l'optimum de la meilleure solution candidate en fonction du temps processeur pour divers jeux d'instances de tailles et de problèmes différents. Une première série de mesures sont effectuées pour un temps processeur accordé de cent cinquante secondes. Pour le PMU, ces mesures sont effectuées au cours de cent essais différents pour chacune des tailles 1000, 2000, 4000, 8000 et 16000. Pour le SISHIM 2, elles le sont au cours d'un seul essai sur un jeu de cents instances différentes pour chacune des tailles 256, 512, 1024, 2048, 4096, 8192 ($k = 2$ et $p = 8,\ldots,13$). Une seconde série de mesures sont effectuées pour un temps processeur accordé de six cents secondes. Pour le PMU, le nombre d'essais est alors réduit à vingt-cinq. Quant au SISHIM 2, c'est le jeu d'instances qui est réduit à vingt-cinq. Ajoutons que, lors de l'utilisation des modules *Gestalt Aléatoire* et *Gestalt AGG*, la taille moyenne des Gestalt variables induites par la meilleure lentille est également relevée.

5.4.5 Résultats et discussion

Les résultats sont présentés et discutés par problème. Aussi, commencerons-nous par ceux obtenus pour le PMU, pour ensuite aborder ceux obtenus pour le SISHIM 2. Nous ne manquerons pas dans ce dernier cas d'effectuer les comparaisons nécessaires avec le PMU afin de discuter, le cas échéant, des différences ou similarités significatives.

5.4.5.1 PMU

Les Figures 5.16, 5.18, 5.20, 5.22 et 5.24 présentent, par taille croissante d'instance de mille à seize mille, la comparaison de l'efficacité du SAG

seul ou assisté d'un des modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*. Les résultats des essais sont résumés sous la forme de boîtes à moustaches décrivant la médiane, les quartiles et les déciles de l'erreur relative Δf de la meilleure solution candidate trouvée en fonction du temps processeur. Les Figures 5.17, 5.19, 5.21, 5.23 et 5.25 présentent, toujours par taille croissante d'instance de mille à seize mille, l'évolution de la médiane des efficacités du SAG seul ou assisté d'un des modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*. Passons en revue les différents résultats et tirons en quelques observations.

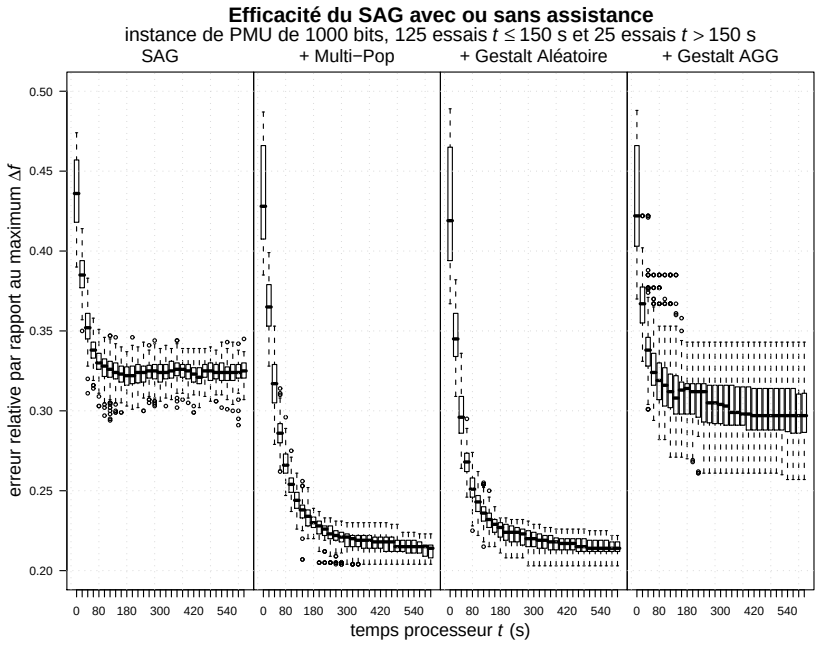


FIGURE 5.16 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 1000 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

Pour le PMU de 1000 bits (voir Fig. 5.16 et 5.17), le module *Gestalt AGG* n'assiste guère efficacement le SAG comparé aux deux autres modules offrant une qualité honorable d'assistance similaire. Ces derniers ne peuvent réellement être différenciés qu'au cours des deux cents premières secondes où le module *Gestalt Aléatoire* permet une convergence légèrement plus rapide. Notons l'importante dispersion de la distribution des résultats obtenus avec le module *Gestalt AGG*.

Pour le PMU de 2000 bits (voir Fig. 5.18 et 5.19), les mêmes observations peuvent être faites que précédemment. La seule différence notable réside

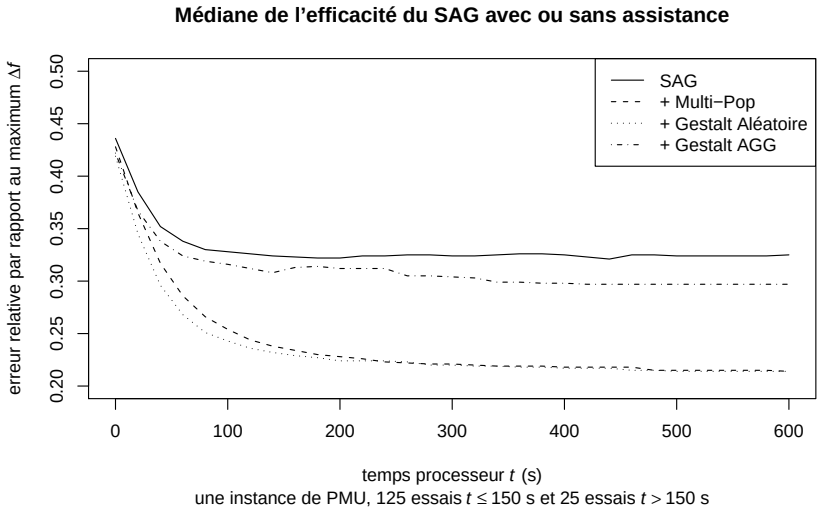


FIGURE 5.17 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 1000 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

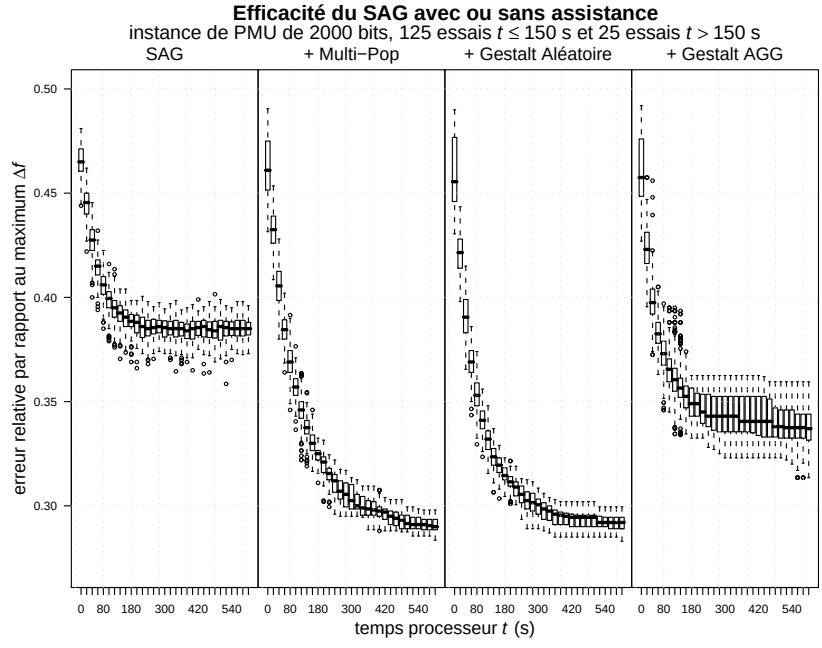


FIGURE 5.18 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 2000 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

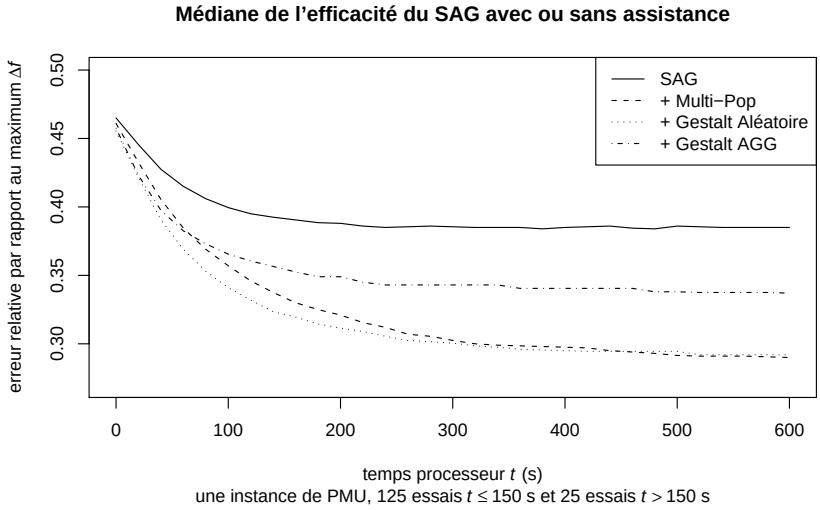


FIGURE 5.19 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 2000 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

dans la légère amélioration de l'assistance prodiguée par le module *Gestalt AGG* par rapport aux deux autres.

Pour le PMU de 4000 bits (voir Fig. 5.20 et 5.21), cette fois le module *Gestalt AGG* rejoint les deux autres modules en termes de qualité d'assistance sans pour autant réellement s'en démarquer. Après deux cents secondes, la meilleure assistance, et de peu, est fournie par le module *Gestalt Aléatoire*.

Pour le PMU de 8000 bits (voir Fig. 5.22 et 5.23), le module *Gestalt AGG* finit par prendre, très légèrement, le dessus en particulier entre cent et quatre cents secondes.

Pour le PMU de 16000 bits (voir Fig. 5.24 et 5.25), nous revenons à un statu quo entre les différents modules. Aucune tendance ne permet de les départager.

Suite à ces observations préliminaires, il est difficile de conclure quoi que ce soit. Tout au plus, nous pouvons affirmer que le SAG a, dans tous les cas, trouvé une assistance positive au moyen d'un des trois modules, autrement dit que le module *Multi-Pop* semble fournir la majorité de la contribution. Nous pouvons également affirmer que le module *Gestalt AGG* a fourni une assistance dont la qualité croît avec la taille de l'instance relativement à celles des deux autres modules, sans pour autant les surpasser. Cette

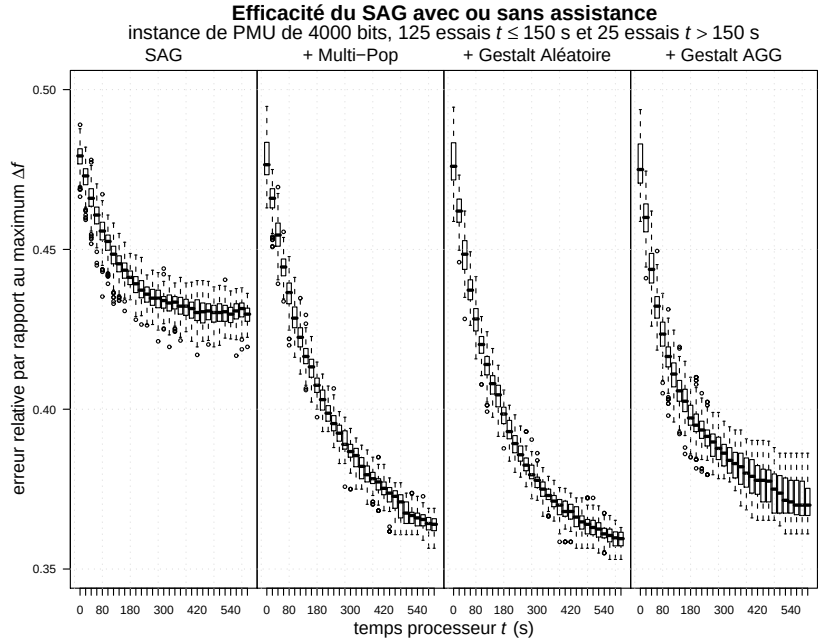


FIGURE 5.20 – Sur cette figure, les assistances du SAG par les trois modules *MultiPop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 4000 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

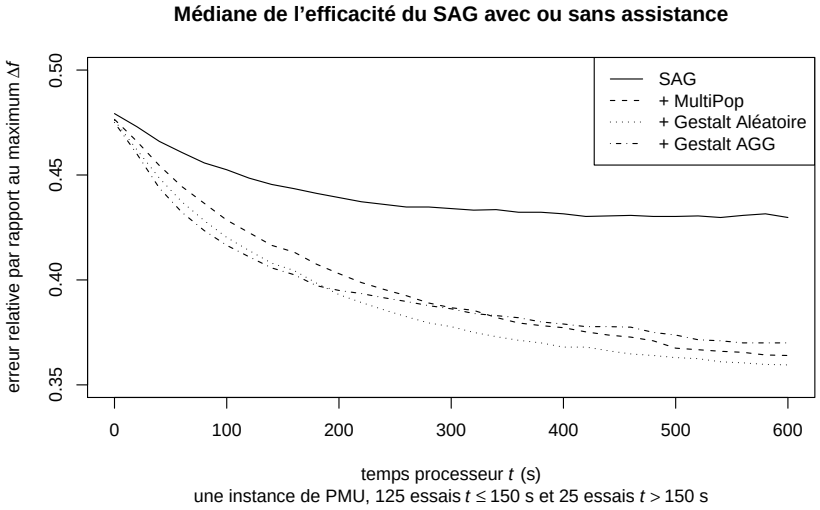


FIGURE 5.21 – Sur cette figure, les assistances du SAG par les trois modules *MultiPop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 4000 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

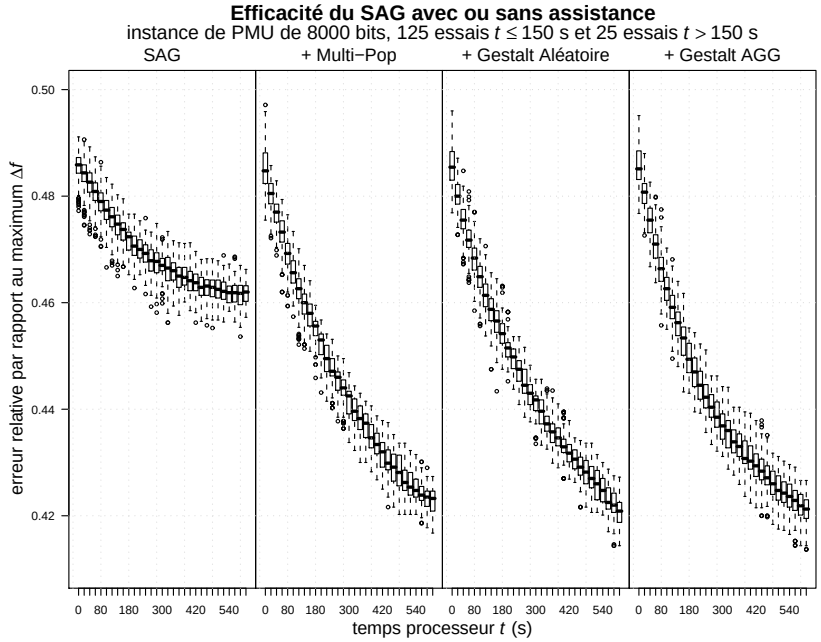


FIGURE 5.22 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 8000 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

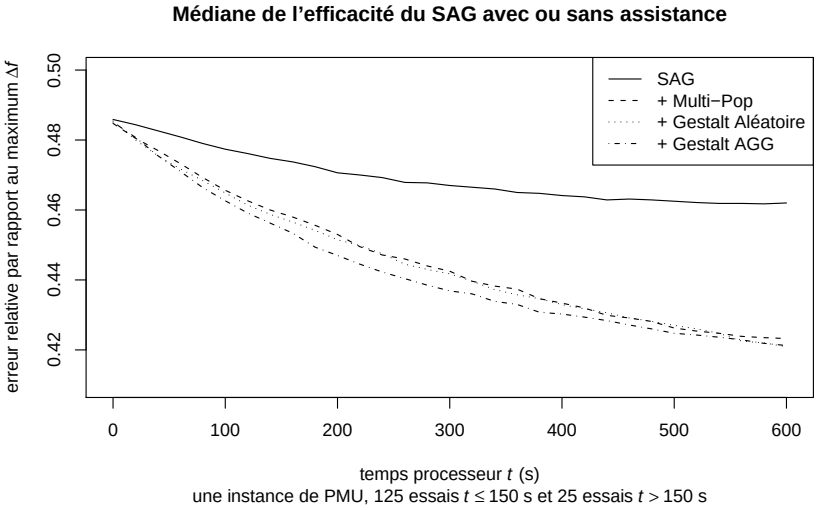


FIGURE 5.23 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 8000 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

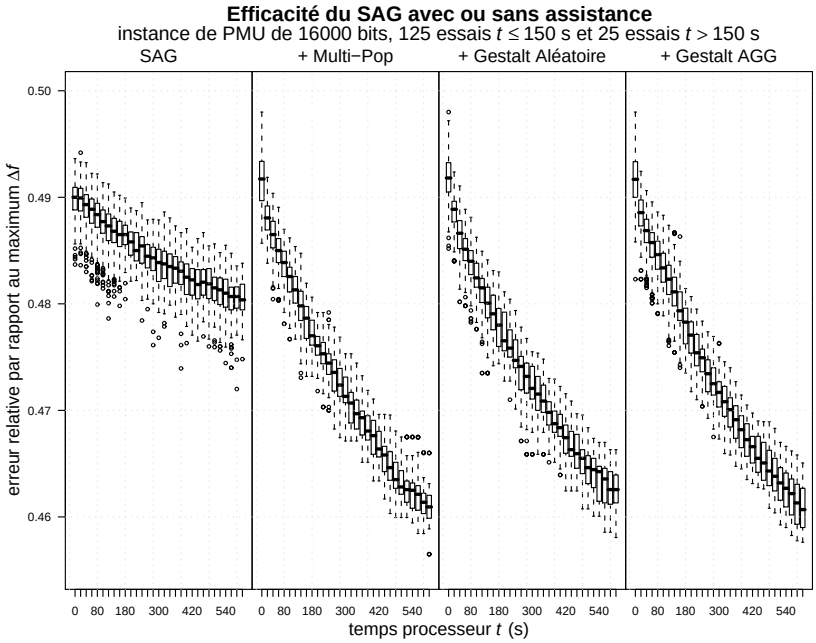


FIGURE 5.24 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 16000 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

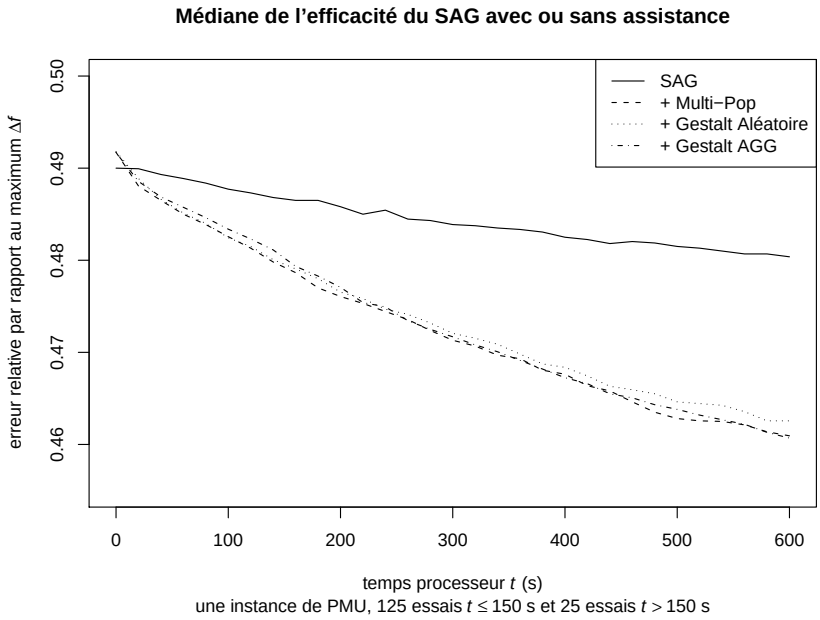


FIGURE 5.25 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour l'instance de PMU de 16000 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

dernière observation pourrait être l'indice confirmant que l'heuristique de la Gestalt permet une meilleure gestion des instances de grande taille.

Afin d'aller un peu plus loin dans cette étude expérimentale, en particulier d'éclaircir cette éventuelle capacité d'échelle, nous allons maintenant comparer les distributions des erreurs relatives Δf en des instants processeurs t bien précis. La taille des instances suivant une progression géométrique de raison deux, nous allons observer les résultats obtenus pour chacune des tailles en des instants suivant une même progression, liant ainsi temps processeur et taille d'instance. Le temps maximum étant de six cents secondes, nous comparerons, pour le PMU de 1000 bits, les distributions Δf du SAG seul ou assité de l'un des trois modules en l'instant $t = 37{,}5$ s. Pour le PMU de 2000 bits, en l'instant $t = 75$ s. Et ainsi de suite, jusqu'au PMU de 16000 bits, en l'instant $t = 600$ s. Tout ceci est présenté au moyen des Fig. 5.26, 5.27, 5.28, 5.29 et 5.30, tandis que les résultats des tests statistiques sont synthétisés au Tableau 5.7.

Au vu des Fig. 5.26, 5.27, 5.28, 5.29 et 5.30 et de l'interprétation statistique reprise au Tableau 5.7, nous pouvons déduire les faits significatifs suivants :

– le SAG est moins efficace lorsqu'il n'est pas assisté,

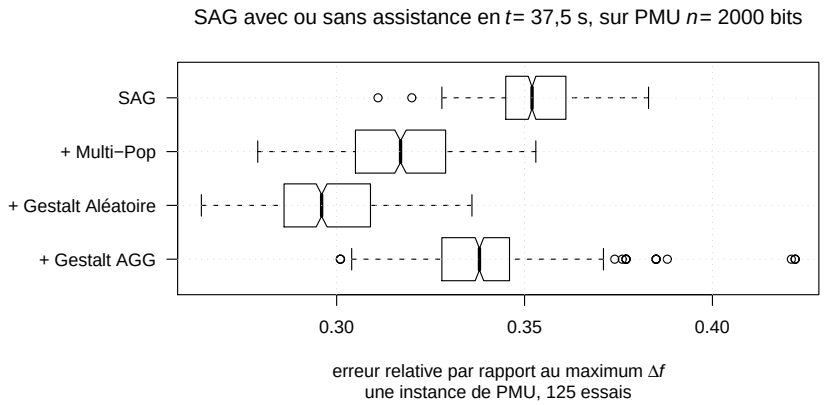


FIGURE 5.26 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le PMU de 1000 bits à l'instant $t = 37,5$ s. Chaque distribution est représentée par une boîte à moustaches.

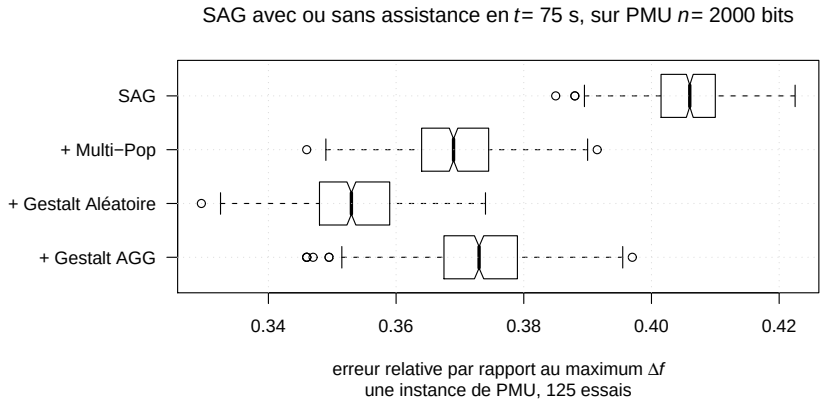


FIGURE 5.27 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le PMU de 2000 bits à l'instant $t = 75$ s. Chaque distribution est représentée par une boîte à moustaches.

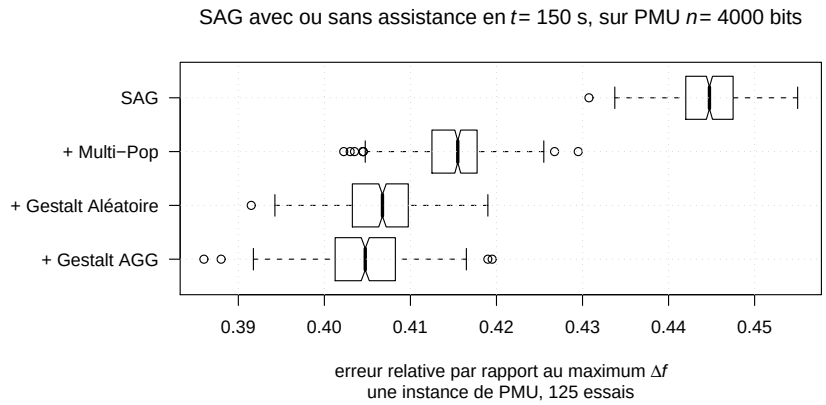


FIGURE 5.28 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le PMU de 4000 bits à l'instant $t = 150$ s. Chaque distribution est représentée par une boîte à moustaches.

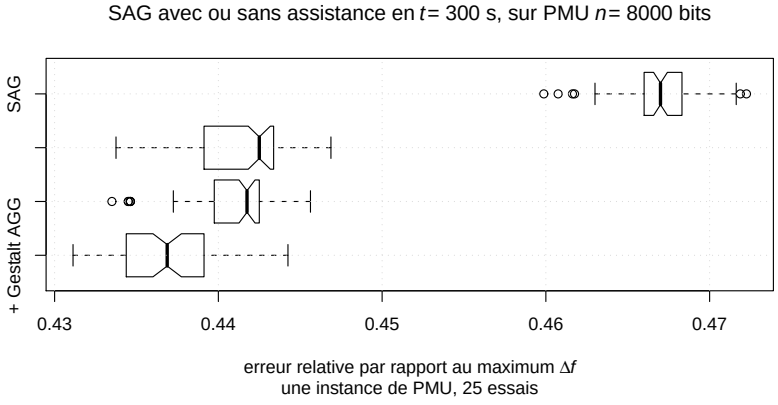


FIGURE 5.29 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le PMU de 8000 bits à l'instant $t = 300$ s. Chaque distribution est représentée par une boîte à moustaches.

p-valeur (Mann-Whitney U à une queue)				
n **(bits)** ; *t* **(s)**	H_1 : **est > que :**	SAG	*Multi-Pop*	*Gestalt Aléa.*
1000 ; 37,5 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	1	1
2000 ; 75 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	1	1
4000 ; 150 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$7{,}305\,10^{-9}$
8000 ; 300 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	0,1854	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$3{,}686\,10^{-12}$	$3{,}172\,10^{-11}$
16000 ; 600 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	1	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	0,4756	$1{,}064\,10^{-4}$

TABLE 5.7 – Le tableau synthétise le résultat des tests statistiques Mann-Whitney U pour comparer, entre eux, les échantillons obtenus par le SAG seul ou assisté des modules *Multi-Pop*, *Gestalt Aléatoire* ou *Gestalt AGG*. Ces tests s'effectuent pour chaque taille d'instance du PMU (indiquée dans la colonne de gauche) en un temps processeur t donné (indiqué également dans la colonne de gauche).Le tableau renseigne les p-valeurs des hypothèses nulles H_0 : la médiane des meilleures solutions candidates obtenues avec x est plus grande que la médiane des meilleurs solutions candidates obtenues avec y, où x est l'un des algorithmes renseignés par les intitulés des trois dernières colonnes et y est un des algorithmes renseignés dans la deuxième colonne. Les p-valeurs sont ajustées avec la méthode de Holm pour trois comparaisons.

- l'assistance fournie par module *Gestalt AGG* s'améliore en fonction de la taille de l'instance et relativement aux autres modules, et
- il est impossible de départager significativement les modules pour le PMU de 16000 bits.

En l'état actuel des résultats, il est difficile de supputer quelque raison que ce soit pour expliquer ce dernier fait, si ce n'est le petit nombre d'essais. Surtout qu'aucune tendance négative ou positive n'est à remarquer à aucun moment donné comme nous pouvons le constater aux Fig. 5.24 et 5.25.

Toutefois, nous pouvons déjà tirer certaines conclusions préliminaires. La réalisation de l'heuristique de la Gestalt en termes du module *Gestalt AGG* démontre des capacités d'assistance positive pour toutes les instances du PMU. La contribution positive de l'évolution des lentilles au moyen de l'AGG s'affirme légèrement et relativement aux autres modules *Gestalt Aléatoire* et *Multi-Pop* en fonction de la taille de l'instance, jusqu'au PMU de 8000 bits y compris.

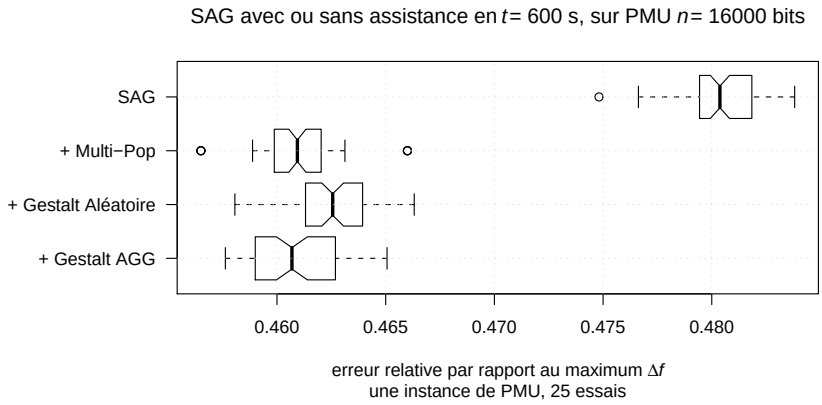


FIGURE 5.30 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le PMU de 16000 bits à l'instant $t = 600$ s. Chaque distribution est représentée par une boîte à moustaches.

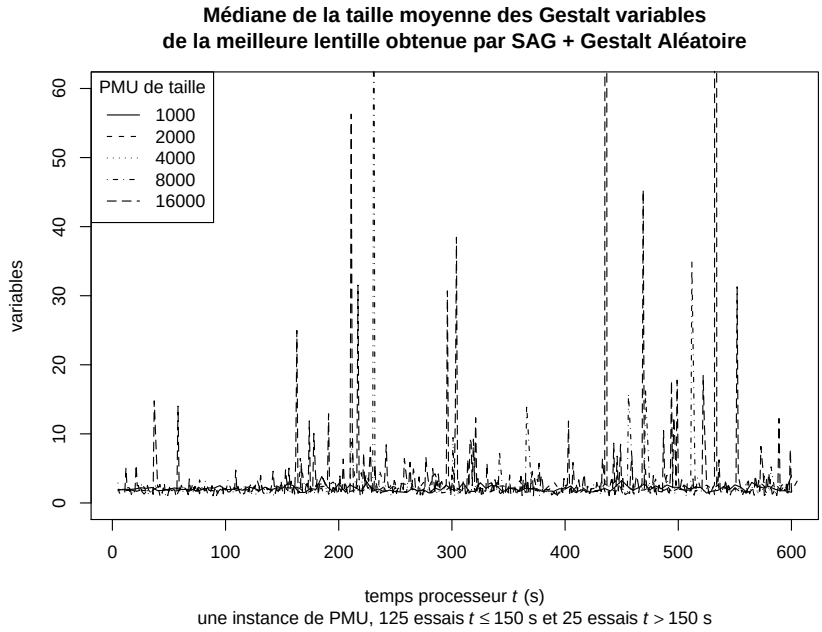


FIGURE 5.31 – Le graphe représente l'évolution, en fonction du temps processeur, de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue au moyen du SAG assisté du module *Gestalt Aléatoire* pour des PMUs de diverses tailles.

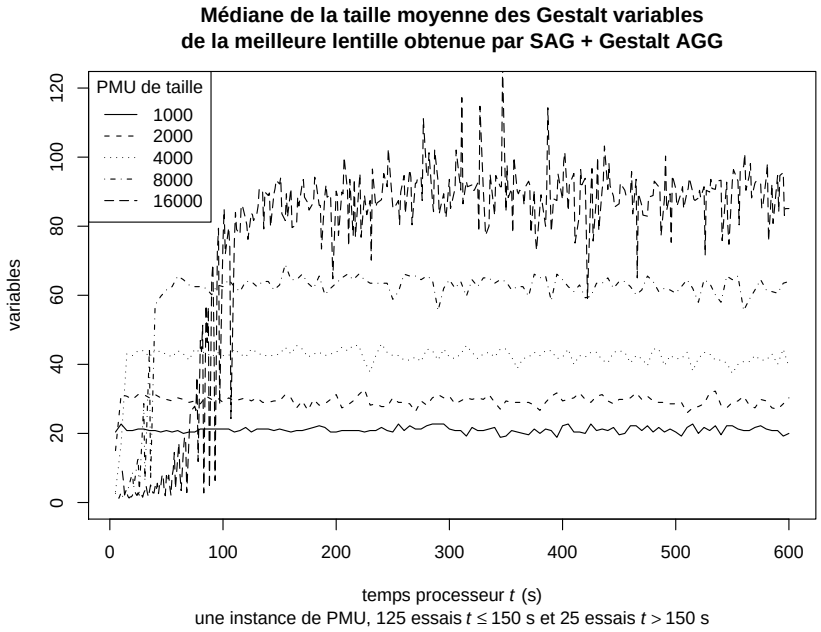


FIGURE 5.32 – Le graphe représente l'évolution, en fonction du temps processeur, de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue au moyen du SAG assisté du module *Gestalt AGG* pour des PMUs de diverses tailles.

Penchons-nous maintenant sur le comportement des modules *Gestalt Aléatoire* et *Gestalt AGG* en termes de taille de Gestalt variables. Les Figures 5.31 et 5.32 reprennent, respectivement pour chacun de ces deux modules, l'évolution de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue en fonction du temps processeur. Pour le module *Gestalt Aléatoire* (voir Fig. 5.31), comme nous pouvions nous y attendre, la taille des Gestalt variables peut varier énormément, quelle que soit la taille de l'instance. Et aucune tendance régulière de leur évolution n'en ressort. Par contre, pour le module *Gestalt AGG* (voir Fig. 5.32), nous pouvons observer une convergence distincte de la taille moyenne des Gestalt variables pour chaque taille d'instance de PMU. Dans tous les cas, les Gestalt variables agrègent un petit nombre de variables au début du processus. Et petit à petit, les lentilles sélectionnées induisent des Gestalt variables de plus en plus grandes. Cette tendance semble ne pas s'inverser de quelque manière que ce soit.

Nous ne pourrons pas aller bien plus loin avec les résultats obtenus pour le PMU. Il nous est nécessaire d'exposer et de discuter les résultats obtenus pour le SISHIM 2 afin d'affiner notre étude expérimentale.

5.4.5.2 SISHIM 2

Les Figures 5.33, 5.35, 5.37, 5.39, 5.41 et 5.43 présentent, par taille croissante d'instance de deux cent cinquante-six à huit mille cent nonante-deux, la comparaison de l'efficacité du SAG seul ou assisté d'un des modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*. Les résultats des essais sont résumés sous la forme de boîtes à moustaches décrivant la médiane, les quartiles et les déciles de l'erreur relative Δf de la meilleure solution candidate trouvée en fonction du temps processeur. Les Figures 5.34, 5.36, 5.38, 5.40, 5.42 et 5.44 présentent, toujours par taille croissante d'instance de deux cent cinquante-six à huit mille cent nonante-deux, l'évolution de la médiane des efficacités du SAG seul ou assisté d'un des modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*. Passons en revue les différents résultats et tirons en quelques observations.

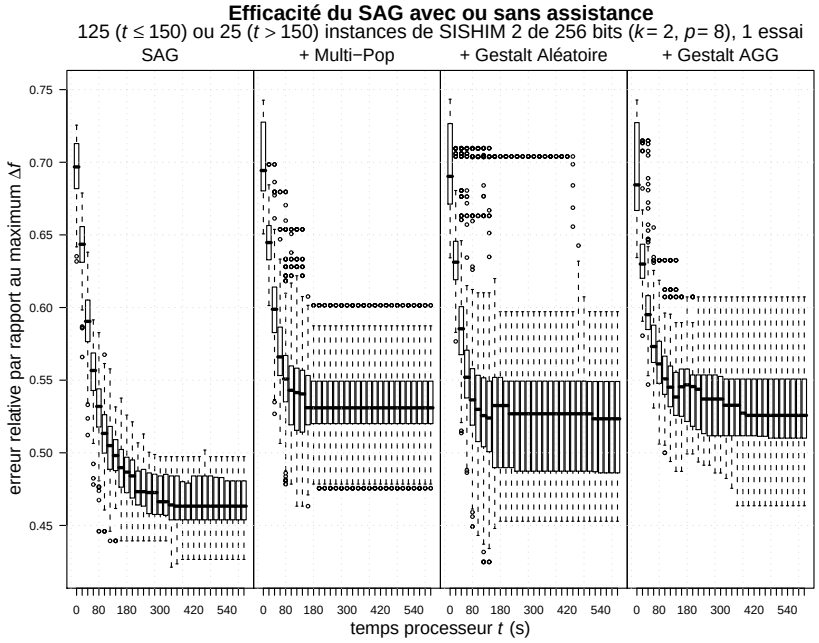


FIGURE 5.33 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 256 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

Pour le SISHIM 2 de 256 bits (voir Fig. 5.33 et 5.34), le SAG n'a nul besoin de l'assistance des modules pour être efficace : les modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* deviennent des obstacles au bon processus du SAG. La Figure 5.33 tempère quelque peu cette observation : au cours

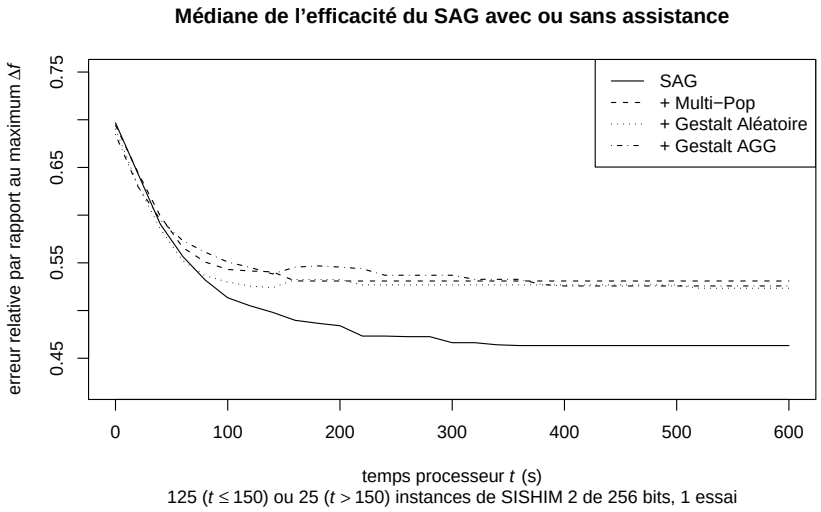


FIGURE 5.34 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 256 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

des cinquante premières secondes, les modules semblent apporter un très léger mieux.

Pour le SISHIM 2 de 512 bits (voir Fig. 5.35 et 5.36), les modules offrent cette fois-ci une assistance positive, en particulier le module *Multi-Pop*. Au cours des premières dizaines de secondes, comme nous pouvons le voir sur la Fig. 5.35, le module *Gestalt AGG* semblerait prendre quelque peu l'avantage.

Pour le SISHIM 2 de 1024 bits (voir Fig. 5.37 et 5.38), l'assistance prodiguée par chacun des modules est désormais bien nette. Les deux modules *Multi-Pop* et *Gestalt Aléatoire* suivent plus ou moins la même tendance. A nouveau, le module *Gestalt AGG* permettrait un meilleur démarrage au cours des cent premières secondes (voir Fig. 5.37).

Pour le SISHIM 2 de 2048 bits (voir Fig. 5.39 et 5.40), de mêmes observations peuvent être faites que précédemment à quelques détails près. Les modules offrent désormais une assistance du même ordre et le module Gestalt AGG démontre à nouveau une pente initiale plus importante pour rejoindre les autres distributions aux environs de trois cents secondes (voir Fig. 5.40).

Pour le SISHIM 2 de 4096 bits (voir Fig. 5.41 et 5.42), cette fois, le module *Gestalt AGG* est le module offrant la meilleure assistance tout au long des

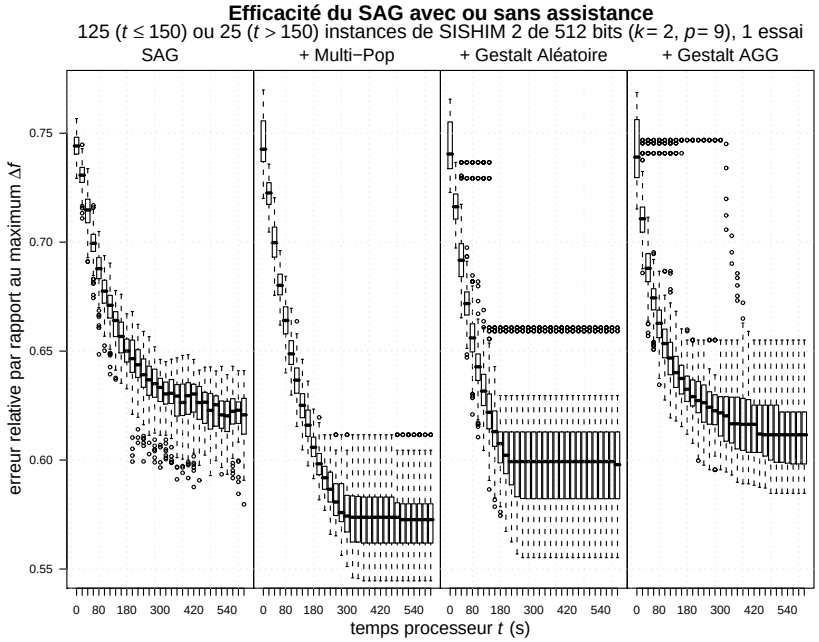


FIGURE 5.35 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 512 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

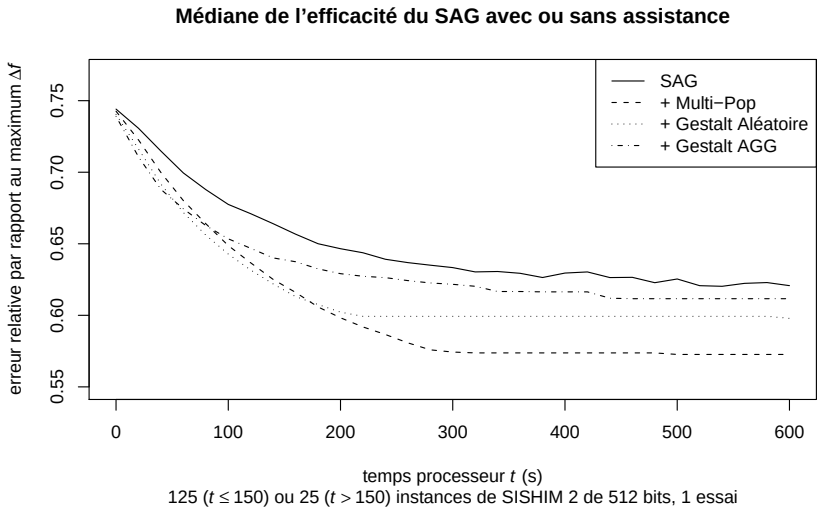


FIGURE 5.36 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 512 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

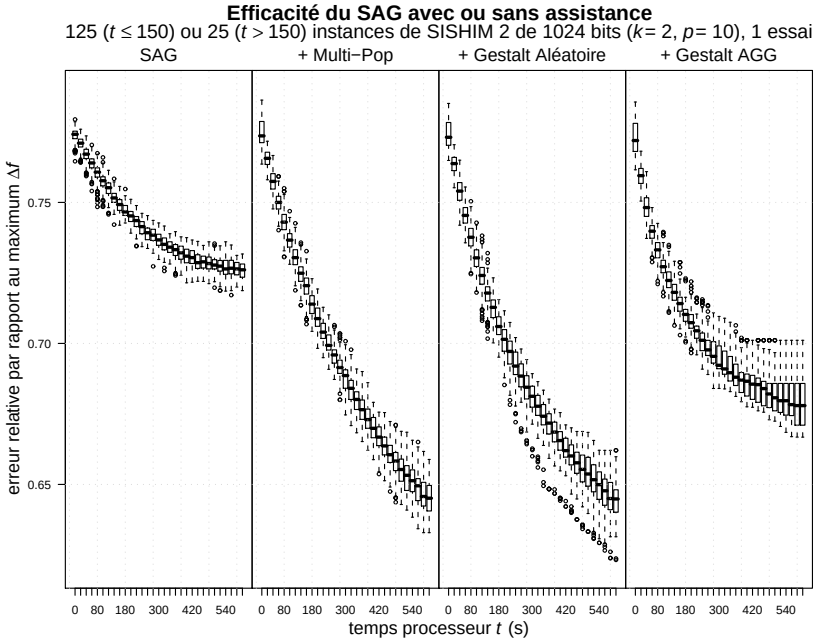


FIGURE 5.37 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM2 de 1024 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

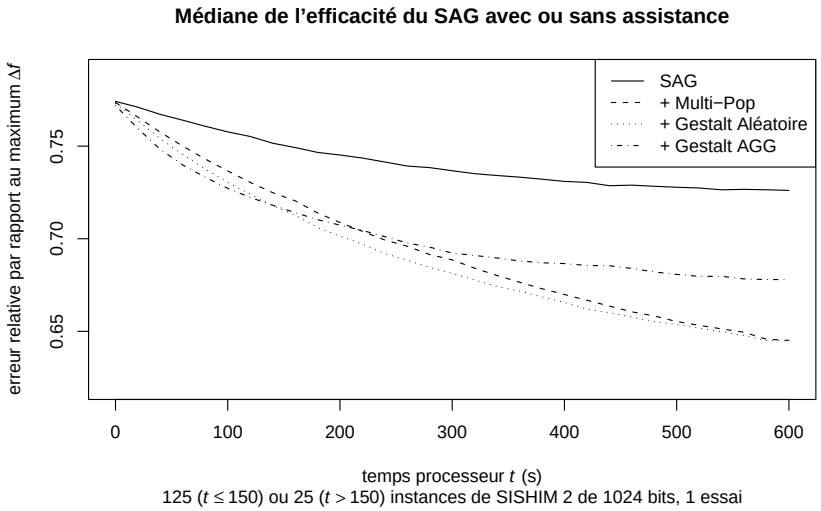


FIGURE 5.38 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 1024 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

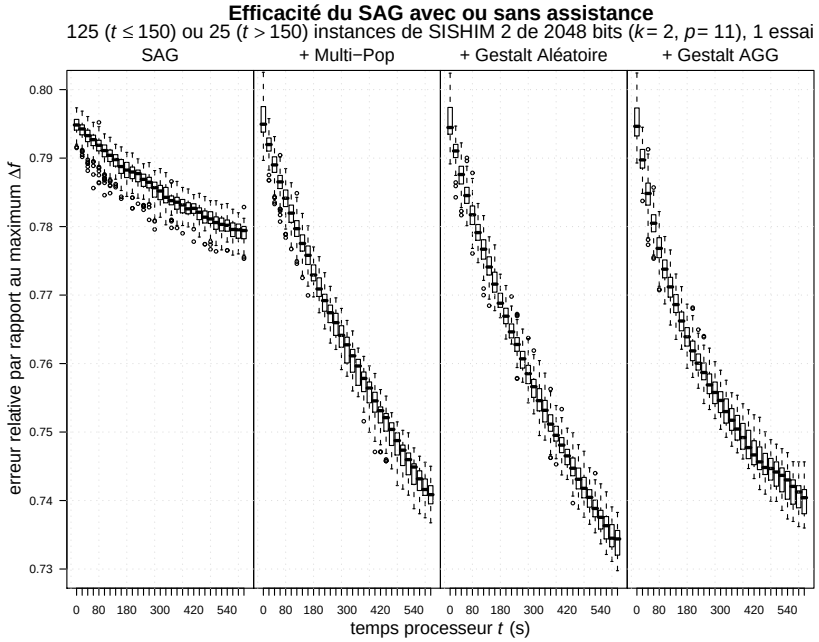


FIGURE 5.39 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM2 de 2048 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

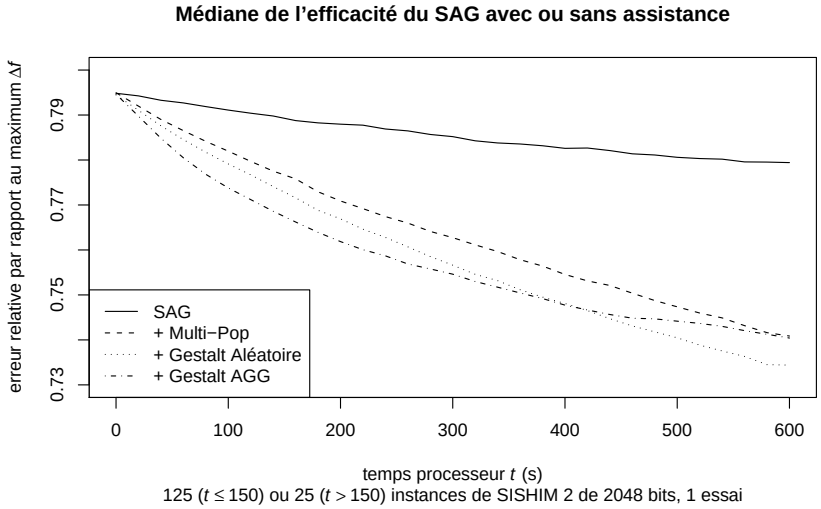


FIGURE 5.40 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 2048 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

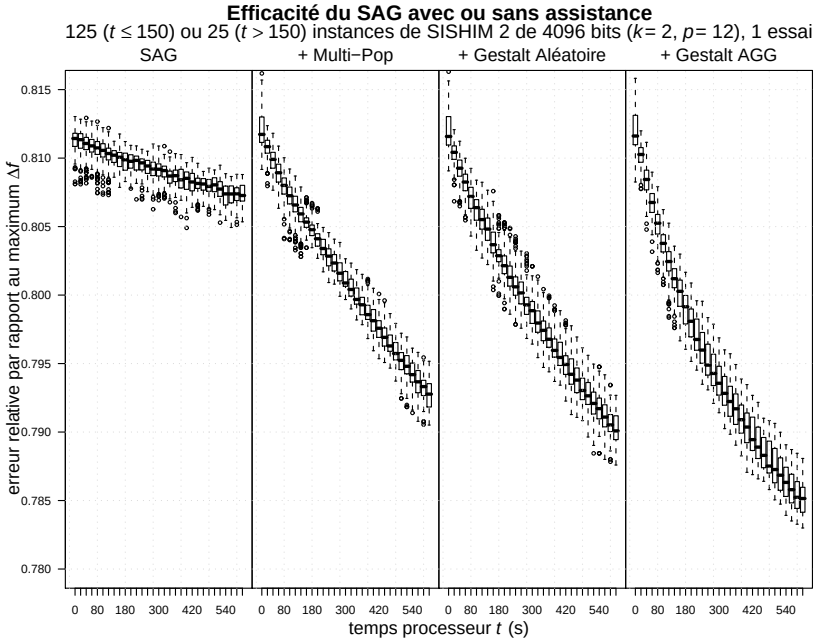


FIGURE 5.41 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM2 de 4096 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

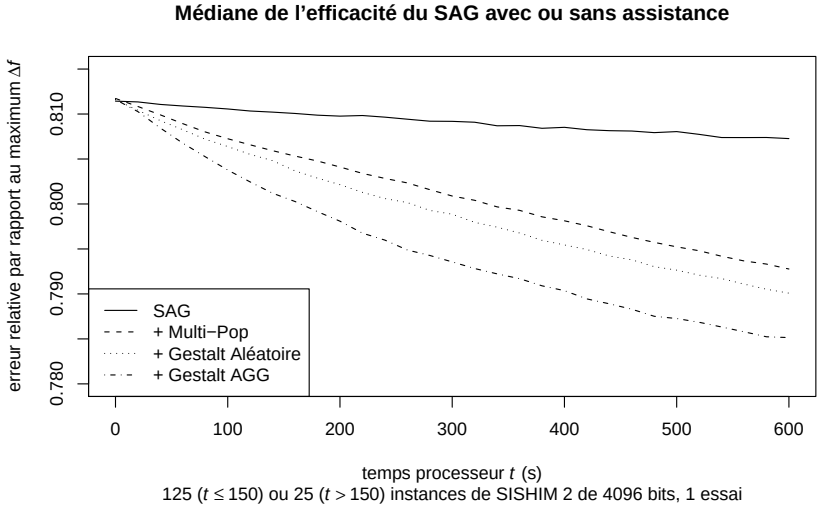


FIGURE 5.42 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 4096 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

six cents secondes. Ensuite, vient le module *Gestalt Aléatoire* suivi de très près de *Multi-Pop* comme le montre la Fig. 5.42.

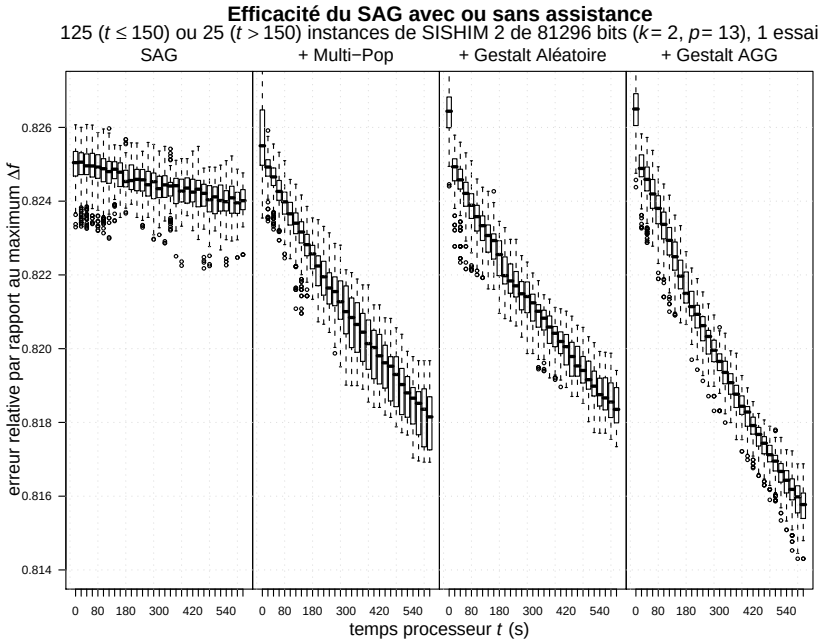


FIGURE 5.43 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM2 de 8192 bits. Pour le SAG et chacun de ces modules, l'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustaches en fonction du temps processeur.

Pour le SISHIM 2 de 8192 bits (voir Fig. 5.43 et 5.44), les observations pouvant être faites sont quasi identiques que pour le SISHIM 2 de 4096 bits.

Suite à ces premières observations, nous pouvons affirmer que l'assistance fournie au SAG s'améliore en fonction de la taille des instances. Les modules bien qu'inutiles pour les instances de 256 bits s'avèrent, au fur et à mesure de la taille des instances, d'une assistance de plus en plus bienvenue. Dans tous les cas, nous avons pu remarquer que la médiane des résultats obtenus au moyen du module *Gestalt AGG* nous indique une meilleure tendance initiale d'assistance, s'amortissant par la suite. Nous ne pouvons dire si ce fait est significatif ou non sur seule base des résultats que nous venons de présenter. Finalement, le module *Gestalt AGG* a fourni une assistance dont la qualité croît avec la taille de l'instance relativement à celles des deux autres modules. Cette observation, nous avions déjà pu la faire pour le PMU à la section précédente. Elle vient donc consolider l'assomption à propos de la capacité de l'heuristique de la Gestalt à démontrer son réel potentiel sur des instances de grande taille.

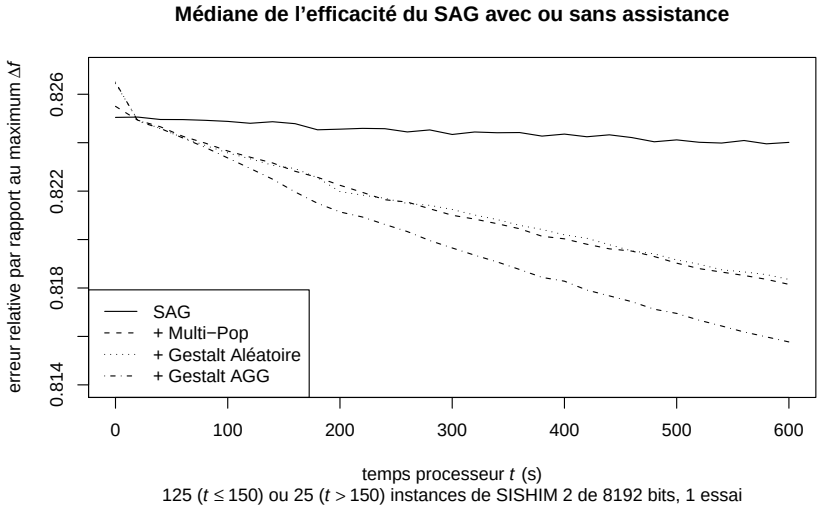


FIGURE 5.44 – Sur cette figure, les assistances du SAG par les trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG* sont comparées pour des instances de SISHIM 2 de 8192 bits. Pour le SAG et chacun de ces modules, l'évolution de la médiane des erreurs relatives des meilleures solutions candidates est représentée en fonction du temps processeur.

Tout comme nous l'avons fait pour le PMU, nous allons maintenant comparer les distributions des erreurs relatives Δf en des instants processeurs t bien précis. Nous allons ainsi observer les résultats obtenus pour chacune des tailles en des instants suivant une même progression géométrique de raison deux que celle formées par les tailles d'instance. Le temps maximum étant de siz cents secondes, nous comparerons, pour les instances de SISHIM 2 de 256 bits, les distributions de Δf du SAG seul ou assisté de l'un des trois modules en l'instant $t = 18{,}75$ s. Pour les instances de SISHIM 2 de 512 bits, en l'instant $t = 37{,}5$ s. Et ainsi de suite, jusqu'à la taille 8192 bits, en l'instant $t = 600$ s. Tout ceci est présenté au moyen des Fig. 5.45, 5.46, 5.47, 5.48, 5.49 et 5.50, tandis que les résultats des tests statistiques sont synthétisés au Tableau 5.8.

Au vu des Fig. 5.45, 5.46, 5.47, 5.48, 5.49 et 5.50 et de l'interprétation statistique reprise au Tableau 5.8, nous pouvons déduire les faits significatifs suivants :

- le SAG est moins efficace lorsqu'il n'est pas assisté,
- l'assistance fournie par le module *Gestalt AGG* s'améliore en fonction de la taille de l'instance et relativement aux autres modules, et
- le module *Gestalt AGG* apporte une assistance significative par rapport aux modules *Gestalt Aléatoire* et *Multi-Pop*.

Ainsi, la réalisation de l'heuristique de la Gestalt en termes du module *Gestalt AGG* démontre à nouveau des capacités d'assistance positive pour

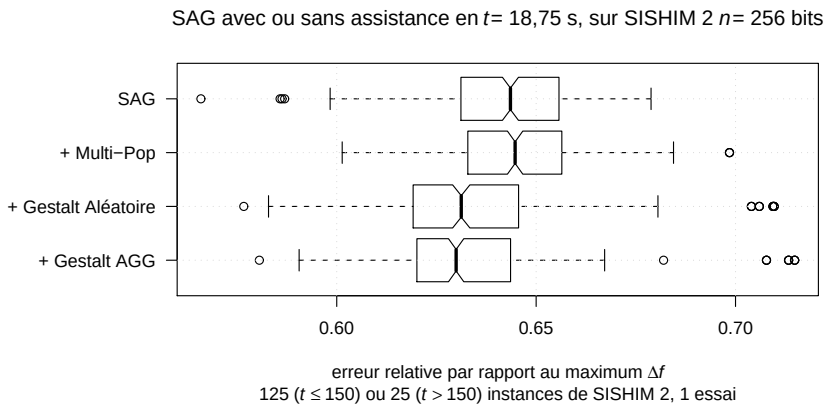


FIGURE 5.45 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 256 bits à l'instant $t = 18{,}75$ s. Chaque distribution est représentée par une boîte à moustaches.

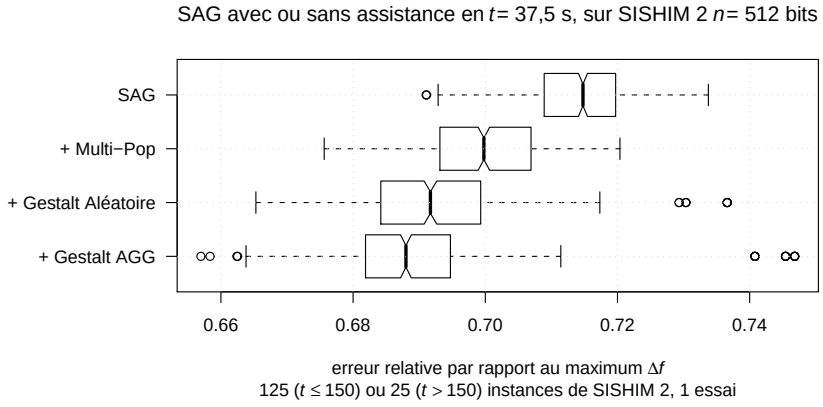


FIGURE 5.46 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 512 bits à l'instant $t = 37{,}5$ s. Chaque distribution est représentée par une boîte à moustaches.

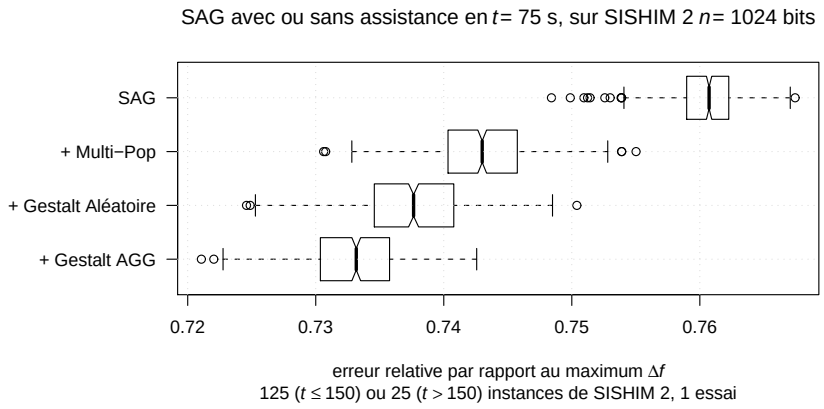


FIGURE 5.47 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 1024 bits à l'instant $t = 75$ s. Chaque distribution est représentée par une boîte à moustaches.

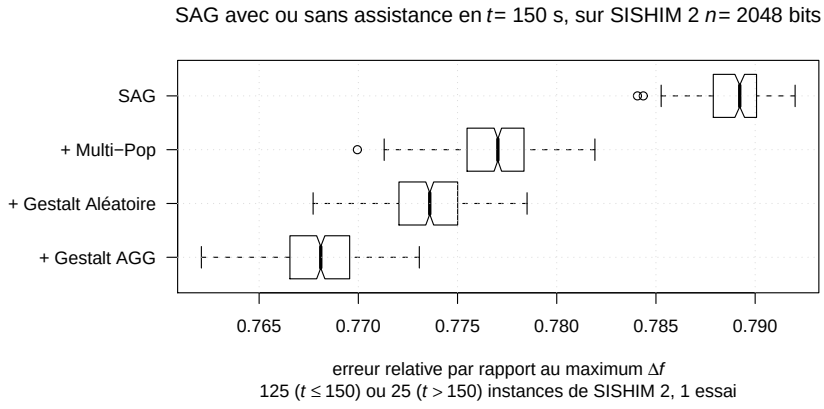


FIGURE 5.48 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 2048 bits à l'instant $t = 150$ s. Chaque distribution est représentée par une boîte à moustaches.

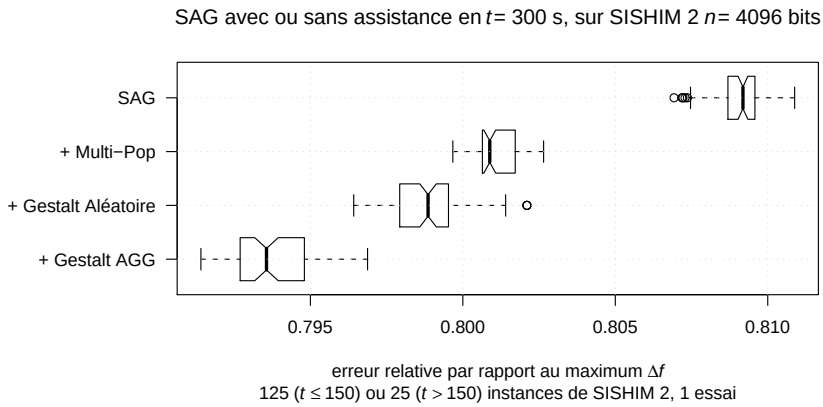


FIGURE 5.49 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 4096 bits à l'instant t = 300 s. Chaque distribution est représentée par une boîte à moustaches.

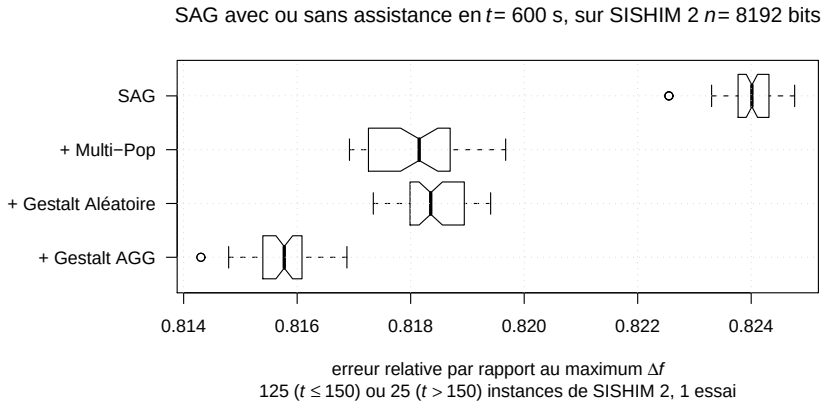


FIGURE 5.50 – Cette figure présente la comparaison de la distribution des erreurs relatives Δf des meilleures solutions candidates obtenues au moyen du SAG seul ou assisté par l'un des trois modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, pour le SISHIM 2 de 8192 bits à l'instant t = 600 s. Chaque distribution est représentée par une boîte à moustaches.

p-valeur (Mann-Whitney U à une queue)				
n **(bits)** ; t **(s)**	H_1 : **est > que :**	SAG	*Multi-Pop*	*Gestalt Aléa.*
256 ; 18,75 s	*Multi-Pop*	0,7829	s.o.	s.o.
	Gestalt Aléa.	$1{,}590\,10^{-13}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	0,2450
512 ; 37,5 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$8{,}697\,10^{-10}$
1024 ; 75 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$
2048 ; 150 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$
4096 ; 300 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$
8192 ; 600 s	*Multi-Pop*	$< 6{,}6\,10^{-16}$	s.o.	s.o.
	Gestalt Aléa.	$< 6{,}6\,10^{-16}$	0,977	s.o.
	Gestalt AGG	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$	$< 6{,}6\,10^{-16}$

TABLE 5.8 – Le tableau synthétise le résultat des tests statistiques Mann-Whitney U pour comparer, entre eux, les échantillons obtenus par le SAG seul ou assisté des modules *Multi-Pop*, *Gestalt Aléatoire* ou *Gestalt AGG*. Ces tests s'effectuent pour chaque taille des instances de SIHIM 2 (indiquée dans la colonne de gauche) en un temps processeur t donné (indiqué également dans la colonne de gauche).Le tableau renseigne les p-valeurs des hypothèses nulles H_0 : la médiane des meilleures solutions candidates obtenues avec x est plus grande que la médiane des meilleurs solutions candidates obtenues avec y, où x est l'un des algorithmes renseignés par les intitulés des trois dernières colonnes et y est un des algorithmes renseignés dans la deuxième colonne. Les p-valeurs sont ajustées avec la méthode de Holm pour trois comparaisons.

toutes les instances du SISHIM 2, et ce de manière plus significative que pour les instances du PMU. La contribution positive de l'évolution des lentilles au moyen de l'AGG s'affirme relativement aux autres modules *Gestalt Aléatoire* et *Multi-Pop* d'autant que la taille de l'instance traitée est grande, démontrant ainsi sa capacité d'échelle. Et à nouveau, ce fait est bien plus marqué que pour les instances du PMU. Rappelons que le PMU et le SISHIM 2 constituent des extrêmes en termes de difficulté. A la lumière de ce rappel, cette différence s'explique d'autant mieux : il est compréhensible que la complexité algorithmique introduite par l'AGG soit quelque peu démesurée par rapport à un problème du type du PMU. D'ailleurs, les modules *Gestalt Aléatoire* et *Multi-Pop* se sont avérés plusieurs fois suffisants à cet égard.

Rappelons également que le SAG et les modules ont été paramétrés pour la plus petite taille des instances traitées. Cette contrainte s'est particulièrement ressentie pour le SAG, ne démontrant que peu d'efficacité pour les instances de grande taille, et ce en particulier pour le SISHIM 2. Le module *Gestalt AGG*, quant à lui, réussit à faire preuve de capacités suffisantes d'adaptation, grâce à des lentilles adéquates, que pour améliorer significativement l'efficacité du SAG.

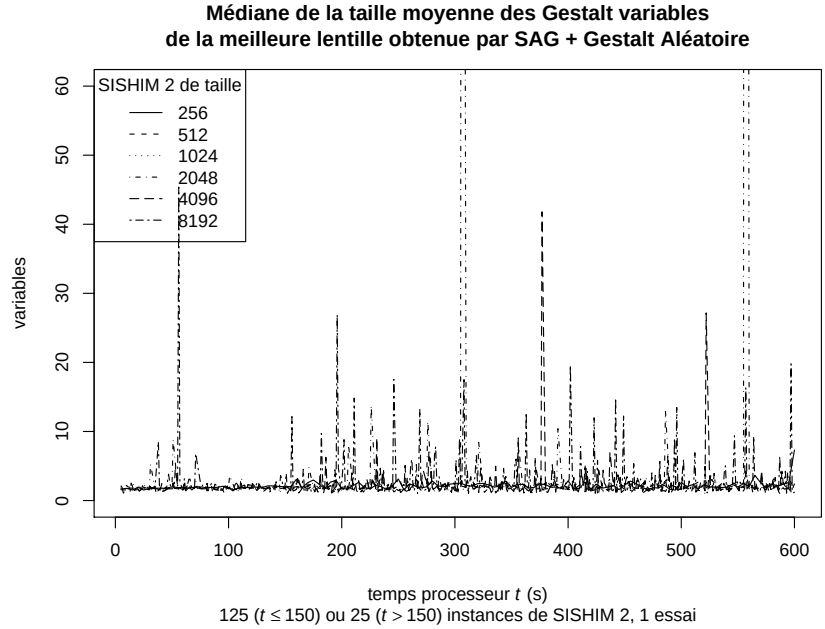


FIGURE 5.51 – Le graphe représente l'évolution, en fonction du temps processeur, de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue au moyen du SAG assisté du module *Gestalt Aléatoire* pour plusieurs instances de SISHIM 2 de diverses tailles.

Nous terminons cette section en étudiant le comportement des modules *Gestalt Aléatoire* et *Gestalt AGG* en termes de taille de Gestalt variables. Les Figures 5.51 et 5.52 reprennent, respectivement pour chacun de ces deux modules, l'évolution de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue en fonction du temps processeur. Pour le module *Gestalt Aléatoire* (voir Fig. 5.51), nous observons le comportement attendu que nous avons également pu constater pour le PMU : la taille des Gestalt variables varie erratiquement et discontinuement, quelle que soit la taille de l'instance. Pour le module *Gestalt AGG* (voir Fig. 5.52), nous observons à nouveau une convergence distincte de la taille moyenne des Gestalt variables pour chaque taille d'instances de SISHIM 2. La tendance observée est fort similaire à celle observée pour le PMU. Toutefois, pour des tailles d'instances similaires, la convergence est plus lente pour le SISHIM 2, mais la médiane de la taille moyenne asymptotique est du même ordre.

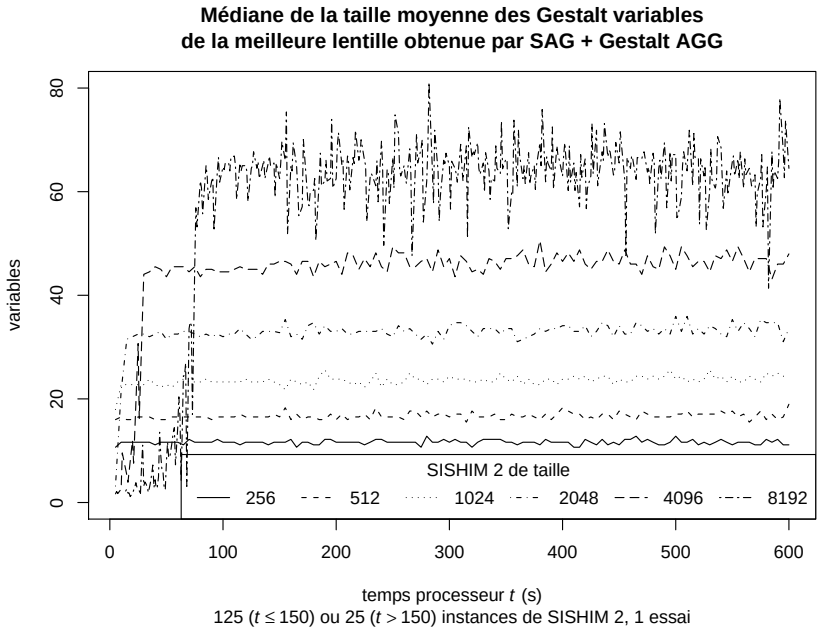


FIGURE 5.52 – Le graphe représente l'évolution, en fonction du temps processeur, de la médiane de la taille moyenne des Gestalts variables de la meilleure lentille obtenue au moyen du SAG assisté du module *Gestalt AGG* pour plusieurs instances de SISHIM 2 de diverses tailles.

D'une part, cela nous indique que l'AGG réagit bel et bien à la complexité du problème. D'autre part, l'AGG tend à ne se comporter que comme un algorithme de groupement. Lorsque l'évolution stagne, il n'y pas de retour en arrière possible, retour qui pourrait s'avérer salvateur en libérant des chemins de recherche alors bloqués par les lentilles.

5.4.6 Conclusion

L'expérience que nous venons de décrire et disserter longuement avait pour objectif d'étudier en détails l'implémentation de l'heuristique de la Gestalt proposée au Chapitre 4. En particulier, les points suivants ont été abordés :

- l'apport des différents composants algorithmiques dans l'assistance prodiguée,
- l'adéquation d'une telle approche à des POCs possédant une structure riche et complexe,
- les capacités d'adaptation, autrement dit la possibilité d'assister effectivement malgré un paramétrage non spécifique,

- les capacités d'échelle, étant entendu l'évolution positive de l'assistance en fonction de la taille de l'instance traitée et relativement à celle fournies par les différents composants algorithmiques, et
- l'étude du mécanisme d'agrégation par l'observation de la taille moyenne des Gestalt variables induites par une lentille.

En considérant l'assistance prodiguée séparément par les modules *Multi-Pop*, *Gestalt Aléatoire* et *Gestalt AGG*, cette dernière constituant la réalisation complète de l'heuristique de la Gestalt, nous avons pu montrer que la construction de lentille constitue un apport significatif dans l'assistance, lorsque le POC considéré comporte une structure riche et complexe à exploiter et que l'instance traitée est de taille conséquente. Ajoutons qu'il a été remarqué qu'au début du processus d'optimisation, cet apport se renforce. L'heuristique de la Gestalt pourrait dès lors être utilisée comme une technique évoluée de préparation ou d'initialisation de population de solutions candidates.

En considérant deux types de problèmes de difficultés différentes, nous avons pu souligner l'adéquation de l'heuristique de la Gestalt à assister des AGs à traiter des problèmes possédant une certaine structure. En ce sens, ceci rejoint le concept de la Gestalt appliquée à l'optimisation qui se veut reformuler la représentation de telle manière à exploiter implicitement l'une ou l'autre propriété structurelle.

En considérant un paramétrage unique et spécifique pour toutes les tailles d'instances considérées, nous avons pu montrer que l'heuristique de la Gestalt permettait à l'algorithme assisté de s'adapter significativement aux tailles supérieures. Ceci pourrait s'avérer particulièrement utile dans un scénario de prototypage rapide où l'on se contentera d'un paramétrage grossier.

En considérant plusieurs tailles d'instance suivant une progression géométrique de raison deux, nous avons pu montrer que l'heuristique de la Gestalt faisait effectivement preuve de capacité d'échelle, en particulier lorsque le POC considéré possède une structure riche et complexe. Ceci supporte l'idée que la Gestalt permet de simplifier le processus de recherche et par là-même d'assister d'autant que la taille est importante.

En considérant l'évolution de la taille moyenne des Gestalt variables dans une lentille, nous avons pu souligner les faiblesses d'une implémentation basée sur l'AGG. Cet algorithme, destiné initialement à des problèmes de groupement, s'acquitte effectivement bien de sa tâche de groupement. Malheureusement, il semble s'y cantonner, empêchant ainsi un dégroupement pouvant s'avérer nécessaire, par exemple lorsque le contexte de recherche nécessite un retour vers la représentation originale pour pouvoir évoluer vers l'optimum.

Ceci termine les diverses expériences menées autour du SAG et des POCs à représentation binaire afin d'étudier l'implémentation de l'heuristique de la Gestalt que nous avons décrite sur cette base au Chapitre 4. Afin d'étendre quelque peu son champs d'assistance, nous allons aborder à la section suivante un tout autre cas de figure.

5.5 Autre problème, autre représentation, autre AG.

Nous venons de boucler un cycle de trois expériences dont les objectifs étaient des preuves de concept de la Gestalt et la validation de l'implémentation de l'heuristique de la Gestalt appliquée au SAG et aux POCs à représentation binaire. Et bien que nous ayons conçu et développé cette heuristique indépendamment de toute métaheuristique, et en particulier AE, et de toute représentation, nous ne pouvons pas étendre de facto nos conclusions expérimentales.

Afin d'illustrer l'application de l'heuristique de la Gestalt à d'autres cas de figure, cette dernière expérience veillera à fournir, par l'exemple, une marche à suivre ad hoc. Pour ce faire, nous considérerons comme autre AG, l'AG d'Ordonnancement (AGO). Cet AG, introduit par Goldberg et al. (1989), dédié aux problèmes de classement, d'ordonnancement, d'agencement ou d'arrangement, dont le PVC est un exemple. Comme POC, nous choisirons d'ailleurs ce dernier, muni d'une représentation basée sur la permutation. Adoptant une attitude réaliste, nous montrerons quelles modifications sont à opérer pour pouvoir appliquer l'heuristique de la Gestalt à l'AGO, quitte à la spécialiser quelque peu.

Contrairement à la précédente expérience, il ne s'agit donc nullement de tester exhaustivement la contribution de l'heuristique de la Gestalt. Nous nous contenterons de comparer l'efficacité de l'AGO avec ou sans son assistance, afin de souligner la réussite de notre adaptation. Pour ce faire, nous considérerons deux types différents d'instances de PVC, plusieurs tailles et deux opérateurs différents, l'un générique et l'autre spécialisé. Cette dernière considération nous permettra de souligner l'intérêt d'utiliser l'heuristique de la Gestalt avec des AEs génériques, avec un minimum de connaissance sur le PVC, sujet avec lequel nous débutons dès à présent la description de l'expérience.

5.5.1 PVC

Le PVC, défini à la Sec. 2.1.2.1 du Chapitre 2, est un POC *NP*-difficile de recherche du plus court chemin passant par un ensemble de villes qui doivent toutes être visitées exactement une fois. Ainsi, considérant le cycle

halmitonien $p := (u_1, \ldots, u_n, u_1)$, où u_i est un sommet de l'ensemble V de n villes, le PVC symétrique consiste à trouver le cycle p minimisant la fonction :

$$f_{\text{PVC}}(p) = w((u_n, u_1)) + \sum_{i=1}^{n-1} w((u_i, u_{i+1})) \ ,$$

où $w : V \times V \to \mathbb{R}^+$ est une fonction réelle positive et symétrique appliquant tout couple de villes sur un nombre réel positif $w(v, v')$ appelé poids ou distance, où v et $v' \in V$. Notons qu'au vu de la définition de f_{PVC}, l'adjacence de deux villes se révèle être une caractéristique importante plus que l'ordre absolu ou relatif selon lequel les villes sont visitées.

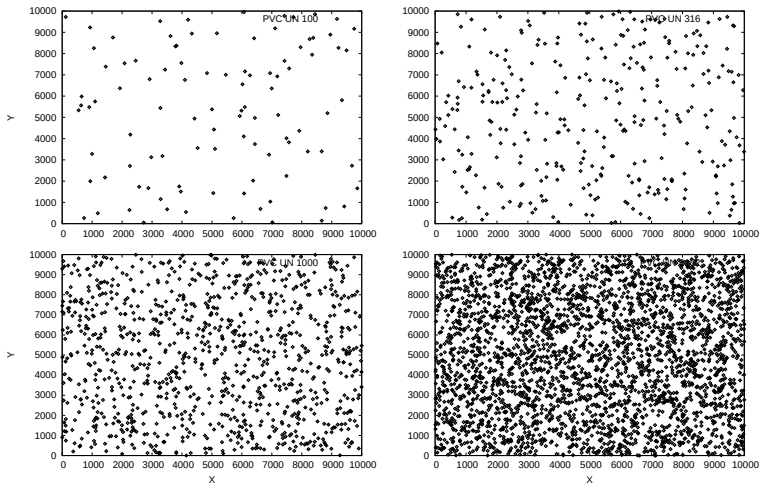


FIGURE 5.53 – Représentation bidimensionnelle de quatre instances de PVC UN de chaque taille 100, 316, 1000 et 3162 (de haut en bas et de gauche à droite).

Lors de la présente expérience, nous considérerons des instances des types uniforme (PVC UN) et en grappe (PVC GR), instances générées aléatoirement au nombre de trente pour une même taille au moyen du générateur de référence TSPLIB [15]. Les instances de type PVC UN se composent de villes dont les positions sont uniformément distribuées dans un carré de dix mille unités de côté en coordonnées bidimensionnelles euclidiennes (voir Fig. 5.53). Les instances de type PVC GR se composent de grappes d'une dizaine de villes dont les barycentres sont uniformément distribués dans un carré d'un million d'unités de côtés en coordonnées bidimensionnelles euclidiennes (voir Fig. 5.54). Pour chacun de ces deux types, nous considérerons quatre tailles formant une progression géométrique de raison $\sqrt{10}$: 100, 316, 1000 et 3162. Nous avons donc huit jeux de trente instances.

15. http://www.research.att.com/~dsj/chtsp/download.html

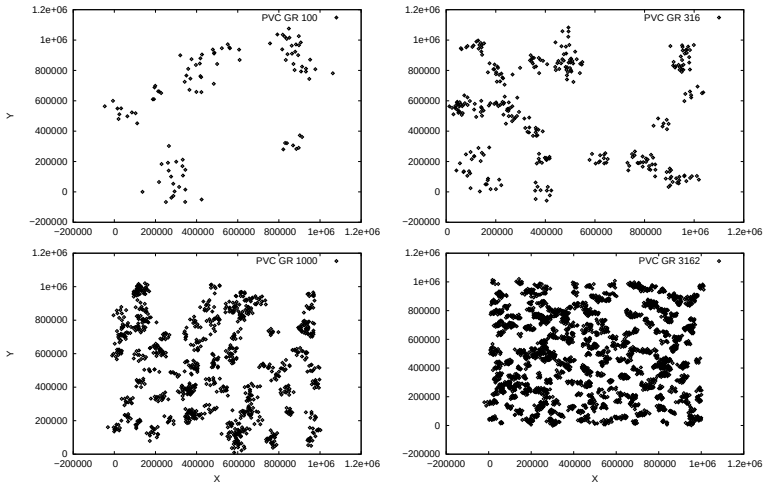


FIGURE 5.54 – Représentation bidimensionnelle de quatre instances de PVC GR de chaque taille 100, 316, 1000 et 3162 (de haut en bas et de gauche à droite).

5.5.2 AGO et son paramétrage

L'AGO, l'AG utilisé au sein de cette expérience et décrit en détails à l'Appendice C, évolue une population sans recouvrement d'individus, dont la représentation se base sur la permutation, au moyen d'une initialisation uniforme, d'une sélection par tournois de deux individus, d'un élitisme [16], d'une mutation par échange de loci adjacents et d'un opérateur de recombinaison tels que le CRA et la Recombinaison par Correspondance Partielle (RCP). Les paramètres de configuration de l'AGO sont la taille de population N_{pop}, la part de la population considérée comme élites N_{eli}, la probabilité de mutation P_μ, et la probabilité de croisement P_X.

5.5.2.1 Représentation du PVC

Une représentation naturelle des solutions candidates d'une instance de PVC est la permutation d'étiquettes de ville, décrivant par là-même un chemin passant par chacune des villes (Goldberg, 1989; Radcliffe and Surry, 1994). Le lecteur intéressé trouvera de plus amples informations à cet égard à la Sec. C.1 de l'Appendice C. Toutefois, permettons-nous d'en rappeler brièvement le principe.

Considérons une instance de PVC de n villes numérotées de 1 à n, constituant l'ensemble des variables $X = \{1,\dots,n\}$. La représentation d'une solution candidate consistera en un n-uplet $p := (p_1, p_2, \dots, p_n)$ où p_1 est fixé arbitrairement à la ville 1, et le $(n-1)$-uplet $(p_2,\dots,p_n)$ est une permutation

16. Une certaine part de la population est copiée dans la nouvelle génération.

sur l'ensemble $\{2,\ldots,n\}$. Le n-uplet p s'interprète alors comme l'ordre dans lequel les villes sont visitées, constituant ainsi un chemin possible, une solution candidate.

5.5.2.2 Opérateurs génétiques de croisement

L'AGO est le pendant du SAG pour les POCs représentés au moyen de permutation. Alors que le SAG utilise exclusivement des opérateurs génériques, l'AGO se voit souvent associé à des opérateurs spécialisés, en particulier ceux de croisement. Afin de traiter le PVC, le choix de l'AG s'est donc naturellement porté sur l'AGO.

Abordons quelques instants les opérateurs de recombinaison que nous utiliserons. La RCP est un opérateur générique et hybride de recombinaison transmettant à la fois l'ordre relatif, la position absolue et l'adjacence présents dans les permutations représentant les individus parents. Le CRA, croisement expressément conçu pour le PVC, exploite exclusivement l'adjacence présente dans les permutations représentant les individus parents.

Rappelons que, conceptuellement, les lentilles n'ont de réel sens que dans l'asssitance d'un algorithme qui n'est pas spécialement conçu pour le problème traité. En exécutant deux versions de l'AGO, l'une utilisant la RCP, l'autre le CRA, nous aurons l'occasion de tester cette hypothèse, et observer, le cas échéant, l'éventuel conflit ou double emploi entre les lentilles et le CRA.

5.5.2.3 Paramétrage

Le paramétrage s'effectue grossièrement et automatiquement, de telle sorte à nous placer dans une situation réaliste. Pour chaque opérateur de recombinaison, la configuration de l'AGO est automatiquement menée par l'algorithme F-Race, présenté au début de ce chapitre, sur l'ensemble des soixante instances PVC UN et PVC GR de taille 100. Un temps processeur de trente secondes est accordé pour chaque configuration testée. La meilleure configuration trouvée sera ensuite utilisée invariablement, quelle que soit l'instance traitée. Au total, nous nous contentons de tester seulement cent quarante-quatre configurations différentes pour chaque opérateur de croisement, produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.9.

Paramètre	Valeurs possibles	Meilleure valeur
RCP		
N_{pop}	{1000,2000,4000}	2000
N_{eli}	{0,1/10,1/20}	1/10
P_{μ}	{0,1/3,2/3,1}	1
P_X	{0,1/3,2/3,1}	1
CRA		
N_{pop}	{1000,2000,4000}	2000
N_{eli}	{0,1/10,1/20}	1/10
P_{μ}	{0,1/3,2/3,1}	1
P_X	{0,1/3,2/3,1}	1

TABLE 5.9 – Le tableau présente, pour les opérateur génétiques de croisement RCP et CRA, les différents ensembles de valeurs pour les paramètres de taille de population N_{pop}, de part de la population considérée comme élites N_{eli}, de probabilité de mutation P_{μ} et de probabilité de recombinaison P_X. Leur arrangement forme une configuration de l'AGO afin de traiter soixante instances de PVC UN et PVC GR de taille 100. Chaque configuration se voit accorder trente secondes de temps processeur.

5.5.3 L'heuristique de la Gestalt : application pratique et paramétrage

Pour assister l'AGO, nous utiliserons l'implémentation de l'heuristique de la Gestalt présentée à la Sec. 4.3 du Chapitre 4. Cette implémentation consiste en un AGG évoluant une population de r lentilles sans recouvrement au moyen d'opérateurs génétiques de groupement (paramétrés par les probabilités de croisement P_X, d'inversion P_I et d'une seule mutation $P_{R\mu}$), d'une sélection EUS, et d'une initialisation uniforme des Gestalt descriptions. Chaque lentille est appliquée à une sous-population distincte de ρ solutions candidates, lui permettant en particulier d'évaluer sa qualité selon une certain fonction d'évaluation g. Cette subdivision de la population en r îles donne lieu à une éventuelle migration entre elles, dont la fréquence t_m et la taille ρ_m peuvent être fixées. Les émigrants sont sélectionnés aléatoirement et remplacent des individus aléatoires dans leur nouvelle population.

Tout ceci régit in fine le mécanisme qui construit de nouvelles lentilles lorsque cela s'avère nécessaire. Il n'est donc nullement affecté par la nature de l'AG [17] assisté. L'interface entre ce dernier et l'heuristique est en effet pleinement réalisée par la lentille et son application. C'est donc à ce niveau que nous devons agir pour pouvoir appliquer l'heuristique de la Gestalt à un AG utilisant une représentation non binaire. Penchons-nous justement

17. Et nous pourrions être plus général en étendant le propos à tous les AEs.

sur la définition de Gestalt description pour une représentation basée sur la permutation, et plus particulièrement celle utilisée pour le PVC.

5.5.3.1 Permutation, Gestalt description et lentille

Rappelons qu'une Gestalt description est une partition de l'ensemble des variables. Intuitivement, si nous reformulons une représentation basée sur la permutation en termes d'une Gestalt description, nous manipulerons, non plus une permutation des variables, mais une permutation des Gestalt variables, autrement dit une permutation de parties de l'ensemble des variables. Dans le cas du PVC, les Gestalt variables peuvent être alors interprétées comme des agglomérations de plusieurs villes, ou des mégapoles. Cette interprétation n'a réellement de sens que si ces villes sont définies comme contiguës, et que cette contiguïté est conservée. Or, l'adjacence, comme déjà mentionné, est une propriété très importante du PVC. Et cette propriété peut s'exprimer très naturellement selon cette interprétation de mégapoles. Voyons ce que cela nous donne formellement.

Considérant l'ensemble de variables $X = \{1, \ldots, n\}$ associé à un domaine commun $\mathbb{D} = X$ et des valuations définies comme des permutations sur X, une Gestalt description G est définie comme un ensemble de m Gestalt variables $\{A, \ldots, M\}$ formant une partition sur X. Ainsi, une Gestalt variable $I \in G$ est un sous-ensemble de X. Les variables i appartenant à I sont alors considérées comme adjacentes, et une valuation de I n'est autre qu'une permutation sur lui-même. Sous une Gestalt description G, une solution candidate peut donc être interprétée comme une permutation de permutation de variables adjacentes. Prenons un exemple. Soit $X = \{1,2,3,4,5,6\}$ et définissons la Gestalt description suivante : $G = \{A,B,C\}$, où $A = \{1\}$, $B = \{3,5,6\}$ et $C = \{2,4\}$. La solution candidate décrite par $(1,4,2,5,3,6)$ peut alors être représentée comme (A,C,B), autrement dit $((1),(4,2),(5,3,6))$, ce qui ne nécessite aucun effort de réinterprétation. Remarquons que nous avons fixé la Gestalt variable A à la variable 1 afin d'éviter une éventuelle dégénérescence de la Gestalt représentation (Radcliffe and Surry, 1994).

Ainsi, sous l'effet de la lentille correspondante, l'AGO ne manipulerait plus des villes mais des mégapoles. Les opérations s'effectuent dès lors sur l'ensemble des mégapoles. La mutation transpose deux mégapoles adjacentes, dans notre exemple si nous transposons B et C, nous obtenons la solution candidate (A,B,C), autrement dit $((1),(5,3,6),(4,2))$. Et comme les Gestalt variables imposent l'adjacence, l'application de la RCP ou du CRA ne pose aucun problème et s'avère des plus naturelles. Dans le cas de la RCP, le jeu de correspondance s'effectue sur les mégapoles. Et dans le cas du CRA, on travaille sur la matrice d'adjacence des mégapoles.

Reste à éclaircir ce qu'il se passe lorsqu'une solution candidate est réinterprétée selon une certaine Gestalt description qui impose des adjacences qui ne s'y trouvent guère. Ce cas de figure arrive en particulier lors d'un changement de lentille. La solution candidate est projetée sur la solution candidate la plus proche respectant la Gestalt description, étant entendu que la proximité s'exprime en termes de transposition. Si nous reprenons notre exemple précédent, et que nous devons changer la Gestalt description pour celle définie par $G = \{A, B, C\}$, où $A = \{1\}$, $B = \{3,5\}$ et $C = \{2,4,6\}$, la solution candidate $((1),(4,2),(5,3,6))$ devient dès lors $((1),(6,4,2),(5,3))$ puisqu'il ne faut qu'une, sans compter 1 qui reste fixé, transposition dans la représentation originelle $(1,4,2,5,3,6)$ pour respecter cette nouvelle Gestalt description.

5.5.3.2 Introduction d'un biais positif quant au PVC GR

L'implémentation de l'heuristique de la Gestalt est basée sur l'AGG. Cet AG utilise souvent dans ses opérateurs de croisement et de mutation l'une ou l'autre heuristique dédiée au problème traité (Falkenauer, 1998). Cette heuristique est utilisée lors de la réaffectation d'un objet à un groupe qui, sans elle, s'effectue aléatoirement. Nous utiliserons cette possibilité lorsque l'heuristique de la Gestalt assistera l'AGO à traiter des instances de PVC GR. Ces instances structurent spatialement les villes en grappe qui en rassemble une dizaine. Nous exploitons cette information de telle manière à ce que la réaffectation d'une ville à une mégapole s'effectue sur base de son voisinage spatial limité aux quinze villes les plus proches.

Choisissons de spécialiser quelque peu l'heuristique de la Gestalt pour plusieurs raisons. Premièrement, l'AGG est à la base prévu pour introduire de telles modifications. L'introduction de ce biais positif est donc relativement simple. Deuxièmement, le coût en termes de ressource n'est pas énorme. Il suffit de construire ce voisinage une seule fois au début de l'optimisation sur base de la matrice de distance. Ceci ne demande qu'un simple tri, et la taille mémoire n'excède pas $60n$ octets, où n est la taille de l'instance. Et troisièmement, ceci rencontre notre souhait d'aborder cette expérience comme un cas pratique. Au vu du peu que cela nous coûte, et du bénéfice que cela pourrait provoquer, il serait, dans un tel scénario, idiot de ne pas en bénéficier. Sans oublier que nous renforçons par là-même la Gestalt qui sera construite.

5.5.3.3 Paramétrage

Nous effectuons, tout comme pour l'AGO, un paramétrage grossier et automatique de l'heuristique de la Gestalt, et ce afin d'assurer l'AGO util-

Paramètre	Valeurs possibles	Meilleure valeur
r	$\{10, 20, 50\}$	10
ρ	N_{pop}/r	200
ρ_m	$\{1/10, 1/5, 1/2\}$	1/2
t_m	$\{10, 20, 50, 100\}$	20 (RCP), 50 (CRA)
P_X	$\{1/4, 1/2, 3/4, 1\}$	1
P_I	$\{1/4, 1/2, 3/4, 1\}$	1
$P_{R\mu}$	$\{1/3, 2/3, 1\}$	1
g	$\{g_{best}, g_{avg}, g_{MPM}\}$	g_{MPM}

TABLE 5.10 – Le tableau présente les différents ensembles de valeurs pour les paramètres de l'heuristique de la Gestalt assistant les AGOs utilisant l'un le RCP, l'autre le CRA et de paramètres $N_{pop} = 2000$, $N_{eli} = 1/10$, $P_\mu = 1$ et $P_X = 1$: la taille de la population de lentilles ou le nombre de sous-populations r, la taille des sous-populations de solutions candidates ρ, la taille de migration ρ_m comme fraction de ρ, la fréquence de migration t_m, les probabilités des opérateurs génétiques de groupement (la recombinaison P_X, l'inversion P_I et la mutation $P_{R\mu}$) et la fonction d'évaluation des lentilles g. Leur arrangement forme une configuration de l'heuristique de la Gestalt afin d'assister l'AGO à traiter un jeu de soixante instances de PVC UN et PVC GR de taille 100. Chaque configuration se voit accorder dix secondes de temps processeur.

isant le RCP ou le CRA. Aussi, pour chacun de ces deux opérateurs de recombinaison utilisés par l'AGO, l'heuristique de la Gestalt est configurée automatiquement au moyen de l'algorithme F-Race, présenté au début de ce chapitre, afin d'assister l'AGO dans sa meilleure configuration (voir le Tableau 5.9) à traiter l'une des soixante instances de PVC UN et PVC GR de taille 100. Un temps processeur de dix secondes est accordé pour chaque configuration testée. La meilleure configuration trouvée sera ensuite utilisée invariablement, quelle que soit l'instance traitée. Au total, cinq mille cent quatre-vingt-quatre configurations sont testées pour chaque opérateur de croisement de l'AGO, configurations produites selon différentes possibilités de valeur pour chaque paramètre, telles que répertoriées dans le Tableau 5.10. Il est un fait intéressant à noter : alors que pour tous les autres paramètres la meilleure valeur trouvée est la même pour le RCP et le CRA, la fréquence de migration diffère selon que l'AGO utilise le RCP et ou le CRA. Elle est plus grande pour ce dernier. Ceci pourrait être le signe que l'assistance de l'heuristique de la Gestalt interfère avec le bon processus du CRA. Les résultats des expériences devraient pouvoir confirmer ou infirmer cette première impression.

5.5.4 Conditions expérimentales

L'expérience présente se veut être l'étude d'un cas pratique. A cet effet, nous nous contenterons de comparer l'assistance prodiguée par l'heuristique de la Gestalt aux AGOs, l'un utilisant la RCP, et l'autre le CRA, afin de traiter des instances de PVC UN et de PVC CR de diverses tailles. Pour ce faire, mesurons l'évolution de l'erreur relative par rapport à l'optimum de la meilleure solution candidate en fonction du temps processeur dans chacun des cas, et ce pour un maximum de cent cinquante secondes de temps processeur. Cette mesure est effectuée au cours de dix essais différents pour chaque type d'instances (PVC UN et PVC CR), chaque taille (100, 316, 1000 et 3162) et chaque algorithme (AGO RCP et AGO CRA) avec et sans assistance. Rappelons que la configuration de l'AGO et de l'heuristique de la Gestalt reste invariablement la même.

5.5.5 Résultats et discussion

Les Figures 5.55, 5.56, 5.57, 5.58, 5.59, 5.60, 5.61 et 5.62 présentent, par ordre croissant de taille d'instance, la comparaison de l'efficacité des AGOs, l'un utilisant la RCP, et l'autre le CRA, seuls ou assistés par l'heuristique de la Gestalt. Les résultats des dix essais sont résumés sous la forme de boîtes à moustaches décrivant la médiane, les quartiles et les déciles de l'erreur relative Δf de la meilleure solution candidate trouvée en fonction du temps processeur. Passons en revue les différents résultats et tirons en quelques observations.

Pour la taille 100 des instances de PVC GR et UN (voir les Fig. 5.55 et 5.56), l'AGO CRA offre, sans grande surprise, la meilleure efficacité in fine. Et l'AGO RCP reste le moins efficace. Lorsqu'ils sont assistés, l'AGO RCP démontre alors une efficacité se rapprochant de l'AGO CRA seul. Alors que l'AGO CRA assisté n'arrive pas à égaler sa version sans assistance. Les AGOs assistés démontrent un meilleur départ. La raison que nous subodorons réside dans l'application des lentilles des premières générations suite à l'initialisation. Nous pouvons remarquer une légère différence entre l'assistance prodiguée à l'AGO RCP pour les instances de PVC GR et pour celles de PVC UN. Dans le premier cas, l'assistance est légèrement meilleure, probablement un effet de la spécialisation ad hoc de l'AGG.

Pour la taille 316 des instances de PVC GR et UN (voir les Fig. 5.57 et 5.58), nous pouvons remarquer de bien plus nettes différences. Cette fois, les AGOs assistés se montrent très tôt efficaces, et sont, en fin de temps processeur, tout aussi compétitifs que l'AGO CRA seul. L'assistance de l'AGO RCP, que démontre l'heuristique de la Gestalt, est à nouveau meilleure dans le cas des instances de PVC GR.

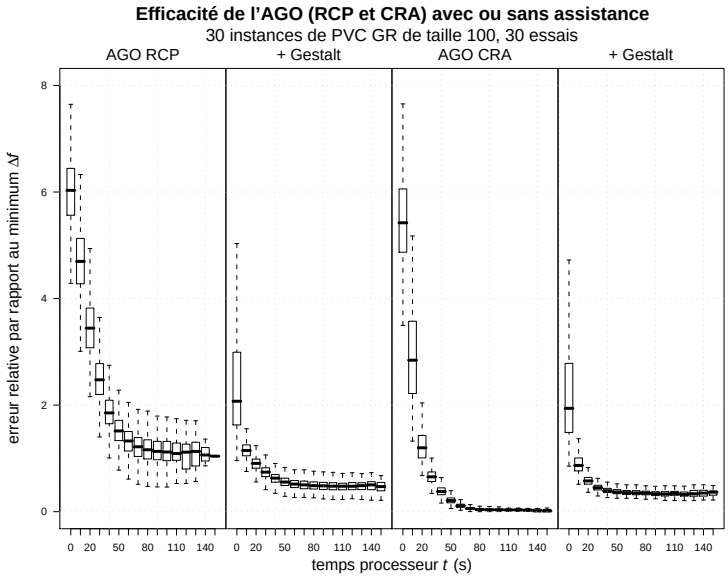


FIGURE 5.55 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC GR de taille 100. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

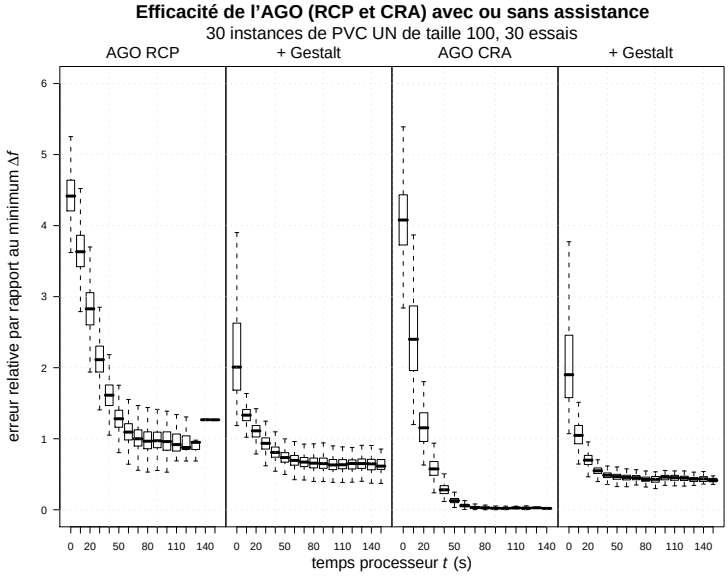


FIGURE 5.56 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC UN de taille 100. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

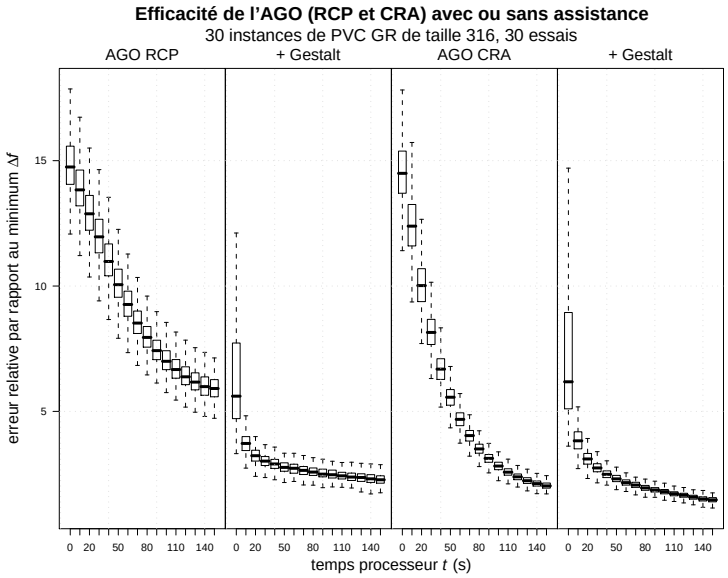


FIGURE 5.57 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC GR de taille 316. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

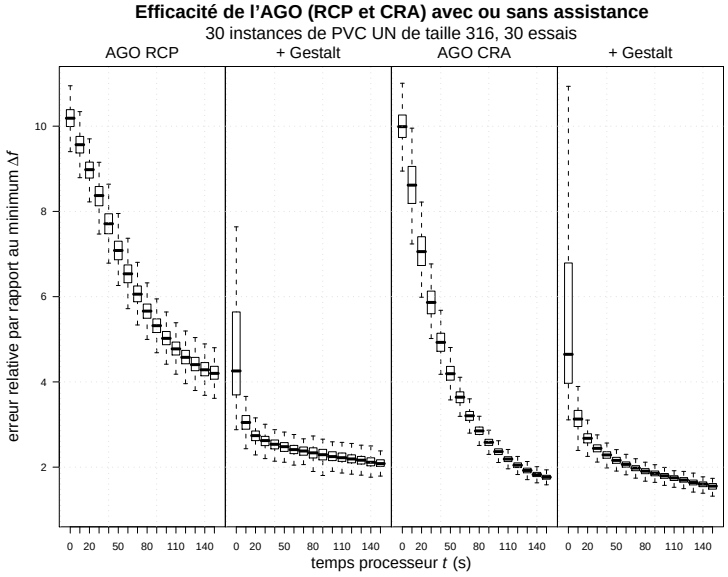


FIGURE 5.58 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC UN de taille 316. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

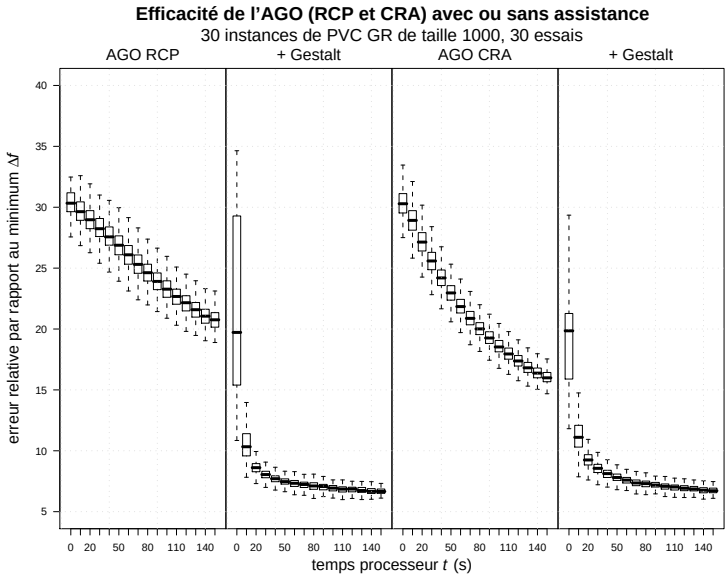


FIGURE 5.59 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC GR de taille 1000. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

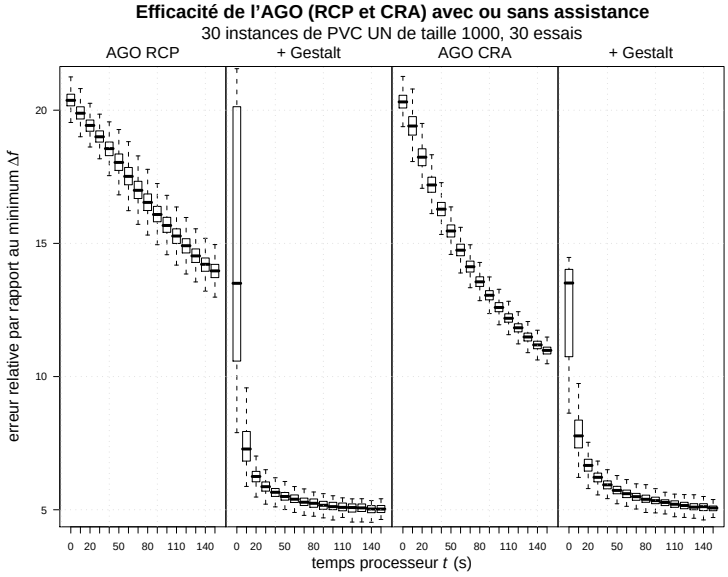


FIGURE 5.60 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC UN de taille 1000. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

Pour la taille 1000 des instances de PVC GR et UN (voir les Fig. 5.59 et 5.60), les AGOs assistés se démarquent significativement et se tiennent dans un mouchoir de poche. Même l'AGO RCP assisté fait mieux à tout niveau que l'AGO CRA non assisté. L'AGO RCP assisté semble converger légèrement plus vite que l'AGO CRA assisté, en particulier pour les instances de PVC GR, et prodigue à nouveau une meilleure assistance pour les instances PVC GR.

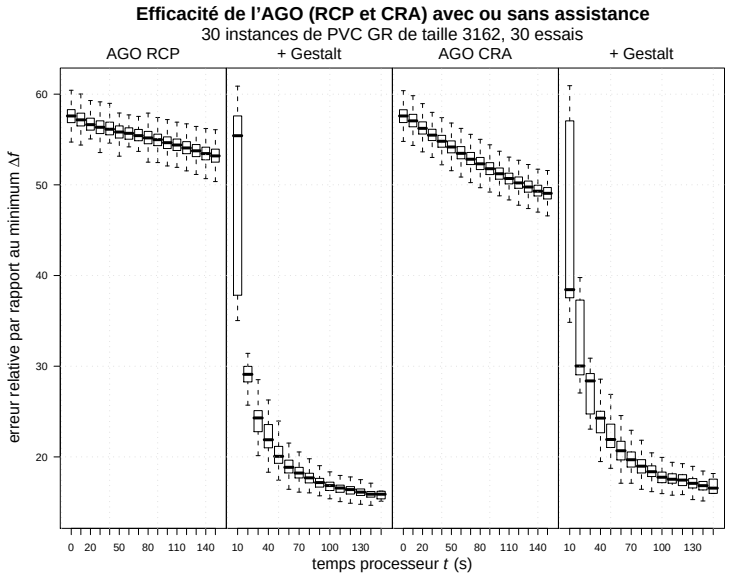


FIGURE 5.61 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC GR de taille 3162. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

Pour la taille 3162 des instances de PVC GR et UN (voir les Fig. 5.61 et 5.62), les observations sont similaires que pour la taille 1000, avec un certain renforcement. En particulier pour l'AGO RCP assisté.

Compte tenu de ces diverses observations, nous pouvons affirmer les faits suivants. L'heuristique de la Gestalt assiste d'autant plus significativement l'AGO, et ce quel que soit l'opérateur de recombinaison, que l'instance est grande. Cependant, nous ne pouvons pas certifier que ces résultats positifs ne sont que la conséquence de l'évolution des lentilles au moyen de l'AGG. Nous avons, par contre, pu relever une différence significative entre les assistances prodiguées à l'AGO RCP selon le type d'instance, différence s'affirmant avec la taille. Nous supputons fortement qu'il s'agit là de la conséquence de l'introduction d'une heuristique dédiée aux instances de type PVC GR dans le mécanisme de l'AGG. Il est également un autre fait remarqué que nous avions prévu : le mauvais couplage de l'heuristique de

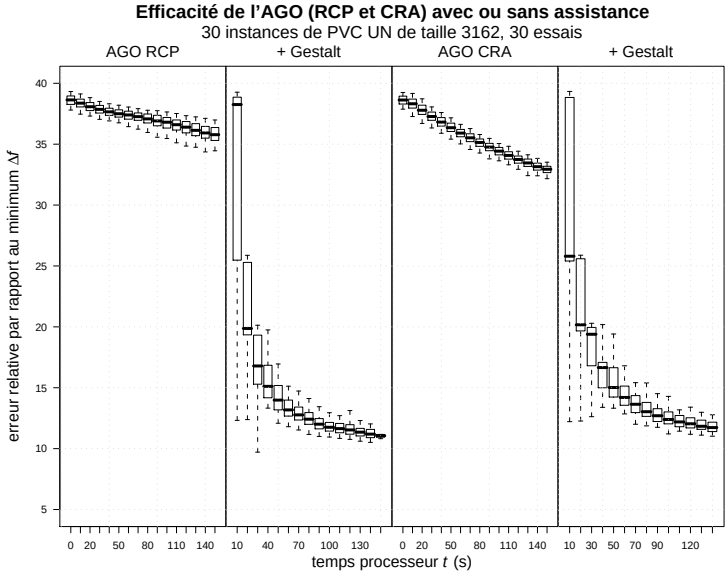


FIGURE 5.62 – Sur cette figure, l'efficacité des AGOs seuls, l'un utilisant la RCP, l'autre le CRA, est comparée avec celle obtenue sous assistance pour les trente instances PVC UN de taille 3162. L'évolution de l'erreur relative par rapport à l'optimum des meilleures solutions candidates est représentée sous forme de boîtes à moustache en fonction du temps processeur.

la Gestalt à l'AGO CRA. Bien que l'efficacité, dont ce dernier fait preuve, est plus qu'honorable, il s'est avéré être moins bien assisté que s'il s'était agi de l'AGO RCP. Et ce quel que soit le type de l'instance traitée. Finalement, lorsque l'AGO est assisté, il fait preuve d'une meilleure convergence, ou de manière plus générale, d'une meilleure tendance, au début de l'optimisation.

5.5.6 Conclusion

L'expérience, dont nous venons de présenter et discuter les résultats, n'avait pour seul but que d'exposer une démarche réaliste dans un cas pratique typique : comment peut-on utiliser pratiquement l'heuristique de la Gestalt ? Le réalisme s'est principalement traduit dans la stratégie de paramétrage et d'application de l'heuristique. Aussi, sommes-nous nous contentés d'un paramétrage relativement grossier et avons-nous, pour les instances de PVC GR, spécialisé quelque peu l'heuristique. Le reste de la démarche s'est résumé à montrer sur quelle partie de l'heuristique fallait-il porter notre attention que pour pouvoir traiter un nouveau type de représentation, et de ce fait d'AG. En somme, nous avons montré comment appliquer pratiquement la philosophie de la Gestalt.

Les résultats expérimentaux n'étaient pas le but premier, bien qu'ils nous aient permis de valider notre approche. Certes, nous ne pouvons affirmer que l'évolution des lentilles est le seul mécanisme responsable. Toutefois, l'heuristique constitue un tout. Sans le modèle de multi-population, une bonne contextualisation ne pourrait être assurée. Nous sommes d'ailleurs d'autant plus confortés dans notre idée que nous avons pu observer un certain nombre de comportements, que nous pourrions qualifier désormais de typiques :

- la bonne accélération initiale,
- le lien entre la qualité relative de l'assistance et la taille de l'instance,
- la difficile interaction avec un opérateur spécialisé, et
- l'impact positif d'une propriété structurelle exprimée implicitement par les Gestalt variables, nous parlons ici bien entendu de l'adjacence dans le cas présent.

Afin de terminer et de conclure cette dernière expérience, émettons encore une remarque. Du fait de sa relative généralité en son propre mécanisme, l'heuristique de la Gestalt constitue un moyen générique et facile d'inoculer de la connaissance a priori dans un AG générique, connaissance qui s'exprimera alors par l'intermédiaire de la Gestalt description et la lentille.

5.6 Conclusion

Pour la première expérience, nous avons pu conclure que les lentilles ont bel et bien un impact sur le processus de recherche mené par le SAG traitant une instance de CRR. Une lentille conçue en respect de la structure du problème, et ce quelle que soit la manière, se constitue effectivement comme un élément d'assistance positive. Néanmoins, l'adéquation de la lentille est une caractéristique sensible dépendant en particulier de l'état actuel de recherche. Ainsi, une lentille statique pourra, au mieux, servir de processus d'initialisation efficace.

Les divers résultats obtenus au cours de la deuxième expérience nous ont démontré l'intérêt d'un mécanisme adaptant les lentilles en cours d'optimisation. Rappelons que l'algorithme assisté est un SAG, et le problème traité un SISHIM. Non seulement le gain par rapport aux résultats obtenus avec le SAG seul est indéniable, mais il est bien meilleur à celui obtenu sous l'assistance de lentilles générées aléatoirement. Bien que nous ayons expressément biaisé les lentilles et leur construction, nous avons ainsi montré qu'il existait un chemin dans l'espace de recherche des lentilles. De plus, ce succès nous montre que les lentilles s'avèrent être un moyen simple et efficace d'inoculer de la connaissance a priori sur un problème par la simple reformulation de sa représentation.

La troisième expérience avait pour objectif d'étudier en détails l'implémentation de l'heuristique de la Gestalt. Nous avons pu montrer que l'évolution de lentille constitue un apport significatif dans l'assistance du SAG, lorsque le POC considéré, le SISHIM 2, comporte une structure riche et complexe à exploiter et que l'instance traitée est de taille conséquente, c'est-à-dire plus grande que deux mille. Ajoutons qu'il a été remarqué qu'au début du processus d'optimisation, cet apport se renforçait. L'heuristique de la Gestalt pourrait dès lors être utilisée comme une technique évoluée de préparation ou d'initialisation de population de solutions candidates. En considérant deux types de problèmes de difficultés différentes, le PMU et le SISHIM 2, nous avons pu souligner l'adéquation de l'heuristique de la Gestalt à assister des AGs traitant des problèmes possédant une certaine structure. En considérant plusieurs tailles d'instances suivant une progression géométrique de raison deux, nous avons pu montrer que l'heuristique de la Gestalt faisait effectivement preuve de capacité d'échelle, en particulier lorsque le POC considéré, le SISHIM 2, possède une structure riche et complexe. Et finalement, en considérant l'évolution de la taille moyenne des Gestalt variables dans une lentille, nous avons pu souligner les faiblesses d'une implémentation basée sur l'AGG. Cet algorithme, destiné initialement à des problèmes de groupement, s'acquitte effectivement bien de sa tâche de groupement. Malheureusement, il semble s'y cantonner, empêchant ainsi un dégroupement pouvant s'avérer nécessaire.

Lors de la quatrième expérience, nous avons pu illustrer la démarche à adopter lorsqu'il s'agit d'appliquer l'heuristique de la Gestalt dans des cas de figure qui n'ont pas encore été rencontrés. Ainsi, avons-nous pu avec succès assister un AGO muni soit de la RCP ou du CRA pour traiter des instances de PVC UN et CR de diverses tailles. Les résultats expérimentaux n'étaient pas le but premier, bien qu'ils nous aient permis de valider notre approche puisque nous avons pu observer une assistance positive. Lors de cette expérience, nous avons à nouveau pu observer un certain nombre de comportements, que nous pourrions qualifier désormais de typiques : la bonne accélération initiale, l'assistance grandissante avec la taille de l'instance, la difficile interaction avec un opérateur spécialisé, et la simple et efficace inoculation d'une propriété structurelle du problème.

Nous pouvons résumer toutes ces observations expérimentales significatives sous la forme de la liste d'assertions suivantes :

1. Conçues en respect de la structure d'une instance de CRR, des lentilles statiques influencent positivement, significativement et pour un certain temps le processus de recherche mené par le SAG traitant une instance du CRR.

2. Spécialisée pour le problème du SISHIM au moyen d'un SAG, l'heuristique de la Gestalt assiste très positivement un autre SAG traitant diverses instances de diverses tailles du problème du SISHIM.
3. L'évolution des lentilles au moyen d'un AGG contribue positivement à l'assistance prodiguée à un SAG. Cette contribution est d'autant plus significative que le paramétrage du SAG assisté n'est pas spécifique aux instances de problème traitées (PMU ou SISHIM 2), que la taille des instances est conséquente, que le problème considéré possède une structure riche et complexe comme le SISHIM 2, et que l'on est tôt dans le processus d'optimisation.
4. L'évolution des lentilles au moyen d'un AGG montre des faiblesses à assister un SAG lorsque le paramétrage du SAG assisté est spécifique aux instances de problème traitées (PMU et SISHIM 2), que la taille des instances est petite, que le problème considéré est simple et linéaire comme le PMU, et que l'on est tard dans le processus d'optimisation.
5. Légèrement spécialisée, l'heuristique de la Gestalt peut être appliquée avec succès à un autre type d'AG comme l'AGO pour traiter diverses instances du PVC. Elle apporte alors une meilleure accélération initiale de la qualité des solutions trouvées et une assistance s'améliorant relativement à la taille de l'instance considérée. Et son interaction avec un opérateur générique telle la RCP est meilleure qu'avec un opérateur spécifique tel le CRA.

Ajoutées des diverses observations faites, notamment de certains comportements spécifiques, nous interprétons ces assertions comme suit :

- l'heuristique de la Gestalt peut se constituer comme une excellente méthode d'initialisation de population de solutions,
- l'heuristique de la Gestalt s'avère un moyen générique et simple d'inoculer de la connaissance à propos du problème dans un AG,
- l'heuristique de la Gestalt se constitue comme une bonne première méta-modélisation dynamique et en ligne pour l'assistance des AGs dans le cadre d'un scénario de la boîte noire, et
- l'AGG ayant démontré ses limitations, la révision de l'implémentation de l'heuristique de la Gestalt pourrait améliorer les performances actuellement obtenues.

6 Travaux apparentés

Après une conséquente étude expérimentale présentée au chapitre précédent, il est temps de positionner quelque peu l'heuristique de la Gestalt par rapport aux travaux qui pourraient, à raison ou non, s'y apparenter. Nous profitons de cette occasion pour lever toutes ambiguïtés et pour débouter toutes mauvais interprétations au sujet de l'heuristique de la Gestalt.

Chaque section est dédicacée à la comparaison avec un sujet particulier, aussi le chapitre se constitue-t-il comme une suite de discussions indépendantes. Toutefois, nous résumerons les conclusions de chaque discussion à la fin du chapitre en guise de sa clôture.

6.1 Algorithmes de recherche à base de modèle

Les algorithmes de recherche à base de modèle constituent une famille de métaheuristiques dont le principe de fonctionnement repose sur l'utilisation et la construction d'un modèle probabilistique paramétrisé (Zlochin et al., 2004). D'une part, le modèle permet de générer des solutions candidates. D'autre part, les solutions nouvellement générées permettent d'affiner le modèle. Plusieurs métaheuristiques appartiennent à cette famille, les plus célèbres étant les algorithmes d'optimisation par colonie de fourmis (Dorigo and Stützle, 2004), les méthodes d'entropie croisée (Rubinstein, 2002), les algorithmes à estimation de distribution (Larranaga, 2002; Pelikan et al., 2002) comprenant l'algorithme d'apprentissage incrémental à population (Baluja, 1994) ou l'algorithme d'optimisation bayésienne hiérarchique (Pelikan, 2005). Certaines de ces métaheuristiques se basent uniquement sur

l'échantillon des solutions actuelles. D'autres utilisent des informations additionnelles collectées durant la recherche et stockées dans une mémoire auxiliaire (Zlochin et al., 2004). Comme le spécifient Zlochin et al. (2004), le terme "modèle" ne désigne nullement une description approximative de l'environnement. Toutefois, il existe une analogie entre le mécanisme adaptatif stochastique et la modélisation de la structure des "solutions prometteuses".

A première vue, l'heuristique de la Gestalt pourrait être considérée comme une méthode de recherche à base de modèle : elle construit une structure en termes de Gestalt variables. Il n'en est rien, et ce pour plusieurs raisons.

Premièrement, l'heuristique de la Gestalt constitue un mécanisme externe ajouté au processus d'une métaheuristique. Observant l'impact qu'elle peut avoir sur cette dernière, l'heuristique structure et adapte la représentation, altérant par là-même le processus de la métaheuristique. Sans métaheuristique, l'heuristique n'a plus de raison d'être.

Deuxièmement, les algorithmes de recherche à base de modèle produisent directement des solutions candidates. L'heuristique de la Gestalt, quant à elle, transforme le processus d'une métaheuristique en reformulant la représentation. En aucun cas, l'heuristique ne produit la moindre solution candidate.

Troisièmement, le mécanisme d'investigation qu'implique l'heuristique de la Gestalt repose plus sur un principe d'apprentissage par renforcement que d'optimisation, et ce notamment par le rôle de la contextualisation de la Gestalt et, a posteriori, sa construction. Il ne s'agit nullement d'expliciter toute mesure effectuée lors de la contextualisation, mais bien de tester ce qui est intrinsèquement important pour la métaheuristique considérée. Et, comme expliqué à la Sec. 3.3, l'heuristique de la Gestalt doit être considérée comme une méta-modélisation de la recherche induite par la métaheuristique utilisée. Elle n'appartient donc pas à la famille des algorithmes de recherche à base de modèle.

6.2 Raffinement multi-niveaux pour problème d'optimisation combinatoire

Walshaw (2004) a introduit, défini et appliqué un nouveau paradigme en optimisation combinatoire. Dans sa plus simple expression, ce paradigme multi-niveaux consiste à construire une hiérarchie d'approximations du problème original. Partant du haut de cette hiérarchie, c'est-à-dire du niveau le plus simple, il s'agit d'y trouver la meilleure solution. Une fois trouvée,

elle est transposée, grâce à un opérateur de projection, dans le niveau inférieur d'approximation, autrement dit raffinée. L'opération est ainsi répétée autant de fois qu'il existe de niveaux d'approximation dans la hiérarchie jusqu'à retrouver le problème original. Walshaw (2004) suggéra clairement le paradigme multi-niveaux comme une fonction additionnelle aux techniques existantes d'optimisation combinatoire. En particulier, il serait plus que justifier d'étendre un algorithme de recherche locale efficace à sa version multi-niveaux.

Un algorithme de raffinement multi-niveaux construit donc une hiérarchie d'approximations du problème original. Une approximations consiste principalement à restreindre l'ensemble des composants à un sous-ensemble. La hiérarchie est alors simplement construite successivement par un algorithme simplifiant le problème à un sous-problème en retirant certains composants. Un algorithme de raffinement, simple recherche locale, s'attellera, lui, à trouver la meilleure solution d'un sous-problème donné. Cette solution optimale d'un certain niveau d'approximation peut alors être étendu à un niveau plus "précis" en rajoutant les composants préalablement occultés. Appliqués itérativement, les algorithmes de raffinage et d'extension finissent par construire une solution candidate complète du problème original. L'efficacité d'un tel algorithme de raffinement multi-niveaux ne s'obtient qu'au prix d'une conception spécifique au problème donné (Walshaw, 2004).

La fonction de l'heuristique de la Gestalt ne se base pas sur une simplification du problème original. L'espace de recherche n'est pas simplifié, seul le graphe de voisinage est modifié. Selon la Gestalt utilisée, une solution candidate peut devenir inaccessible, bien que présente dans l'espace de recherche. Cette nuance peut paraître anodine, et pourtant, elle est lourde de conséquences. L'évaluation d'une solution candidate s'effectue en effet toujours selon le problème original et il n'y a aucune transformation à opérer pour l'obtenir. L'heuristique agit donc sur les possibilités d'action de la métaheuristique et ne transforme nullement le problème en des versions simplifiées. Cette stratégie reste très générique et ce quelque soit le problème traité. Il ne s'agit que de l'agrégation de variables d'une représentation. Ainsi, et ce malgré des apparences similaires, l'heuristique de la Gestalt n'est pas un type d'algorithme de raffinage multi-niveaux.

6.3 Recherche à voisinage variable

La recherche à voisinage variable, un cadre générique de recherche locale introduit par Mladenović and Hansen (1997), est principalement basée sur un changement dynamique du voisinage. L'algorithme canonique d'une

telle stratégie considère ainsi différentes relations de voisinage, induisant des voisinages emboîtés de différentes tailles. Ordonnant les voisinages par leur taille et partant du voisinage le plus petit, il y itère une recherche locale jusqu'à trouver un optimum local. Si l'algorithme ne parvient pas à atteindre un optimum local au bout d'un certain nombre d'itérations, l'algorithme choisit le voisinage de la taille supérieure pour y entamer sa recherche locale. Si, suite à ce changement, un optimum local est trouvé, l'algorithme revient alors au voisinage précédent. Dans le cas contraire, le voisinage de la taille supérieure est choisi et ainsi de suite. Différentes versions de ce type d'algorithme existent et se différencient en la définition des relations de voisinages et des stratégies d'échange de voisinages.

Comme expliqué dans la Sec. 4.1.2, l'heuristique de la Gestalt modifie dynamiquement la relation de voisinage, l'objectif étant d'assister le processus de la métaheuristique. Cette modification est dirigée selon sa qualité d'assistance, aussi faut-il évaluer son impact, bénéfique ou négatif, sur la recherche de bonnes solutions candidates menée par la métaheuristique. L'heuristique de la Gestalt se base donc sur une stratégie de changement des relations de voisinage. Au vu de ces caractéristiques fonctionnelles, elle pourrait être classée comme une recherche à voisinage variable. Toutefois, de subtiles différences l'en séparent quelque peu, différences que nous décrivons ci-après.

Premièrement, l'heuristique proposée n'est pas un algorithme complet d'optimisation combinatoire, mais une stratégie générique pour assister une métaheuristique dans sa tâche d'optimisation. Deuxièmement, la construction des lentilles, qui modifie le graphe de voisinage, ne dépend pas directement de la découverte d'un optimum local, mais dépend plutôt du bénéfice apporté lors de la recherche d'un optimum. Troisièmement, malgré la possibilité de diminution de la taille d'un voisinage, cette diminution de voisins n'empêche nullement l'augmentation de la distance entre voisins (voir Sec. 4.1.2), contrairement à la recherche à voisinage variable qui repose en particulier sur la diminution de cette distance. Quatrièmement, l'apprentissage mené au sein de l'heuristique de la Gestalt ne dépend pas de la conception de relations de voisinage spécifiques au problème traité. En conclusion, l'heuristique de la Gestalt et la recherche à voisinage variable se limitent à partager un même type de mécanisme d'adaptation de voisinage, mais ce mécanisme n'est pas conduit selon les mêmes objectifs.

6.4 Algorithmes évolutionnistes

Au cours de cette thèse, une implémentation évolutionniste s'est imposée comme choix naturel pour supporter l'heuristique de la Gestalt (voir

Sec.4.3). De par leur support évolutionniste, les principes développés dans le cadre de l'heuristique de la Gestalt pourraient apparaître comme similaires à des concepts importants tels que l'hypothèse des blocs constitutifs ou *building blocks*. Afin d'éviter toute ambiguïté et incompréhension, nous allons clarifier certains concepts sous-jacents de l'heuristique de la Gestalt.

6.4.1 Schemata

Le théorème du schema, introduit par Holland (1975), est la première tentative théorique de fournir un modèle de la dynamique des AGs [1]. Ce modèle supporte certaines de leurs capacités et tente de prédire la qualité de la génération suivante. Cependant, le théorème du schema ne permet pas d'expliquer certains comportements observés et d'autres scientifiques, par exemple Vose (1998), ont fourni de meilleurs modèles ou l'ont étendu, par exemple à tout type de génotype comme Radcliffe (1991).

Ce théorème est construit sur le concept de *schema* : un motif identifiant un sous-ensemble de chromosomes similaires selon certaines gènes. Le *schema* est ainsi défini par un ensemble d'allèles constants de certains des gènes dans le chromosome. Formellement et considérant un chromosome de n gènes $g_i \in \{0,1\}$, le schema est un n-uplet de ternaires $(h_1,\ldots,h_n)$ où $h_i \in \{0,1,\#\}$ et le symbole #, appelé "don't care", détermine un allèle indéterminé. Par exemple, le schema $(0,1,\#,\#)$ détermine l'hyperplan, l'ensemble de tous les vecteurs de bits suivant : $(0,1,0,0)$, $(0,1,0,1)$, $(0,1,1,1)$ et $(0,1,1,0)$.

Une Gestalt description consiste en des groupes disjoints deux à deux de variables, les Gestalt variables. Leur définition n'implique pas la définition d'un ensemble d'allèles constants. Par conséquent, une Gestalt description ne correspond pas à un schema.

6.4.2 Hypothèse des blocs constitutifs et apprentissage de lien

Un bloc constitutif, ou *building block* en anglais, est un groupe d'allèles dont les positions sont rapprochées (Holland, 1975). Ce concept est le coeur de l'hypothèse du même nom, une théorie ayant pour objectif d'expliquer la dynamique adaptative du SAG. Cette théorie, suggérée pour la première fois par Holland (1975) et inspirée par la théorie de l'évolution sexuelle de Fisher (Fisher, 1958), décrit principalement un mécanisme adaptatif identifiant et recombinant des blocs constitutifs. Cette hypothèse est, encore de nos jours, discutée et sujette à controverses. Il n'est en effet toujours pas évident que de tels blocs constitutifs soient identifiés par le SAG. Pour exemple, des problèmes hiérarchiques tels que les fonctions "Royal Road"

1. Les chromosomes sont alors encodés par des vecteurs de bits.

(Mitchell et al., 1992) furent spécialement conçus pour tester l'hypothèse des blocs constitutifs, en particulier leur assemblage hiérarchique. Cet essai expérimental se conclût par un échec du SAG et déboucha sur l'élaboration d'algorithmes totalement dédicacés à la découverte et l'apprentissage de lien, c'est-à-dire des relations entre gènes ou la proximité de gènes dans des blocs constitutifs. Parmi ces algorithmes, peuvent être cités l'AG brouillon (Goldberg et al., 1989), l'AG compact (Harik et al., 1999) ou l'algorithme d'optimisation bayésienne hiérarchique (Pelikan, 2005). L'heuristique de la Gestalt se différencie de ces algorithmes de par la non-manipulation de blocs constitutifs et de son mécanisme fondamental.

Dans l'heuristique de la Gestalt, les Gestalt variables sont définies comme des groupes de variables. Dans le cadre des algorithmes évolutionnistes, elles peuvent être alors considérées comme des groupes de gènes, mais à ne pas confondre avec des blocs constitutifs. Un bloc constitutif définit en effet des configurations allèliques spécifiques de gènes, autrement dit des configurations valuées précises d'un groupe de variables.

Les algorithmes d'apprentissage de lien fondent leur processus sur la découverte de dépendances entre variables valuées. L'heuristique de la Gestalt, quant à elle, construit une Gestalt description sur base de l'efficacité du processus de recherche de la métaheuristique assistée dans l'optimisation de l'instance de problème considérée. Les objectifs de ces deux approches sont différentes : la première vise à décrire l'instance de problème en termes de liens de dépendance entre variables alors que la seconde est conçue pour assister le processus d'une métaheuristique donnée en altérant la représentation du problème. Il n'est en effet pas important pour l'heuristique de la Gestalt de découvrir la vraie relation de dépendance. Seule l'efficacité de la reformulation du problème en termes d'une Gestalt dans l'assistance de la métaheuristique choisie importe.

6.4.3 Evolution compositionnelle

Watson (2006) définit l'évolution compositionnelle comme un processus évolutionniste impliquant la combinaison de systèmes ou de sous-systèmes de matériel génétique pré-adapté [2] semi-indépendamment. Dans cette métaphore biologique, une solution complexe résulte alors de l'interaction de parties de solutions, parties évoluant au moyen de ces mêmes interactions. Plusieurs algorithmes se fondent sur ce concept d'évolution compositionnelle tels que le modèle d'adaptation évolutionniste symbiogénique (Watson, 2002) et l'algorithme de transition évolutionniste (Defaweux, 2006).

2. La pré-adaptation désigne essentiellement une coïncidence entre les modifications héréditaires fortuites de l'espèce et la disponibilité d'une niche écologique accessible aux individus qui la composent.

Tous deux usent de deux idées clefs dans l'évolution compositionnelle : l'identification des bonnes collaborations par relations symbiotiques et l'agrégation de solutions partielles en solutions complexes.

L'heuristique de la Gestalt n'est pas un mécanisme construisant des solutions candidates à une instance de problème, mais un algorithme d'apprentissage dédicacé à assister des algorithmes d'optimisation dans leur tâche d'optimisation. L'heuristique transforme la représentation d'un problème en fixant une structure particulière. Cette structure n'induit nullement la division d'une solution candidate en parties.

L'implémentation évolutionniste, que la présente thèse suggère, repose entre autres sur l'utilisation du modèle multi-îles afin de pouvoir évaluer les lentilles (cf Sec. 4.3.4). Ce modèle s'avère induire une évolution compositionnelle sous certaines conditions (Skolicki, 2007). La configuration impliquant de nombreuses petites îles échangeant aléatoirement des individus remplit ces conditions et donne lieu à une coopération entre les îles. Des parties de solutions sont alors échangées entre îles en recombinant les individus émigrés et locaux. L'implémentation choisie pour l'heuristique de la Gestalt, et telle qu'elle a été appliquée à des AEs, transforme donc les AEs originaux en AEs compositionnels. Remis dans le contexte de l'heuristique de la Gestalt, cette transformation peut prendre sens. En effet, rappelons que l'un des principes importants de son mécanisme est la contextualisation. D'une certaine manière, l'évolution compositionnelle permet d'entrechoquer, d'échanger et de diffuser les contextes pertinents pour l'évaluation des lentilles.

6.5 Hyperheuristique

Le succès d'une métaheuristique dépend non seulement du choix de représentation mais également des opérateurs. Leur combinaison génère un graphe de voisinage qui sera plus ou moins adapté à l'instance du problème traité. Et la conception dédicacée de ces deux aspects transformera la métaheuristique en un algorithme spécifique au problème considéré, voire à toute une famille de problèmes. Mais cet effort de conception ad hoc coûte en temps de développement, un coût qui pourrait être amorti si cette tâche était quelque peu automatisée. C'est ce que proposent les hyperheuristiques (Cowling et al., 2001). Ces nouvelles heuristiques explorent principalement un espace de recherche d'opérateurs avec pour objectif d'obtenir des métaheuristiques adaptatives. L'heuristique de la Gestalt peut être considérée comme un membre de cette famille, comme nous allons l'expliquer dans la suite. Mais avant d'arriver à la conclusion, il nous faut revenir quelque peu sur la notion d'opérateurs.

La plupart des opérateurs de métaheuristiques sont stochastiques et peuvent être définis comme une fonction n appliquant s sur un $s' \in n(s)$ avec une certaine probabilité de transition $p_{ss'} = P(s'|s)$. Toutes ces probabilités de transition ne sont pas données explicitement, mais dérivées de lois probabilistes paramétriques utilisées dans l'algorithme de l'opérateur. Parfois, elles sont également gouvernées par certaines mesures telle que l'évaluation de la solution candidate. Prenons, par exemple, la mutation du SAG (voir Appendice A) : le basculement d'un bit. Cet opérateur bascule chaque bit d'un vecteur de bits, le chromosome, avec une probabilité donnée comme un paramètre de taux de mutation $\mu \in [0,1]$. Si le chromosome est long de l bits, et la distance de Hamming entre s et s' est $m \in [0,l]$, la probabilité $p_{ss'}$ est alors égale à $\mu^m(1-\mu)^{l-m}$.

Comme expliqué dans Sec. 3.2.1, la modification de la représentation induit une modification du graphe de voisinage ν, et donc des opérateurs et du processus de la métaheuristique. Si la représentation est modifiée, la probabilité de transition $p_{ss'}$ est alors biaisée. En effet, une altération, telle qu'une Gestalt, d'une représentation peut supprimer ou créer certaines transitions de voisinage. Cette action tout-ou-rien peut ainsi être un moyen de guider le processus de recherche.

L'heuristique de la Gestalt altère la représentation et biaise les opérateurs stochastiques utilisés. Compte tenu de la définition des hyperheuristiques (Cowling et al., 2001), l'heuristique de la Gestalt peut être considérée comme une hyperheuristique explorant l'espace de recherche des opérateurs induits par le biais de $p_{ss'}$.

6.6 Emergence intrinsèque

Comme nous l'avons expliqué dans la Sec. 3.2.3, une Gestalt se décrit et se construit comme une structure de variables sous la forme d'agrégations. La représentation se voit alors reformulée sous la forme de motifs macroscopiques abstrayant la description original et les opérateurs. Cette abstraction biaise et transforme alors le graphe de voisinage, et de ce fait, la manière dont la recherche est conduite par la métaheuristique, afin d'assister cette dernière. La métaheuristique se voit, en quelque sorte, alors dotée de nouvelles capacités. Ce processus entier semble constituer une forme d'émergence selon la définition de Bersini (2004); Crutchfield (1994); Shalizi (2001) : l'émergence intrinsèque.

Selon Crutchfield (1994), la formation de motifs, quoi que jouant un rôle fondamental, n'est pas une caractéristique suffisante pour constituer un phénomène d'émergence de comportement coordonné et de traitement global informationnel. Crutchfield (1994) introduit alors la notion

d'*émergence intrinsèque*. Un tel phénomène s'avère pouvoir se résumer à la formation de motifs procurant des fonctionnalités additionnelles de traitement global informationnel au système. Ceci donne lieu à une augmentation des capacités computationnelles intrinsèques. Shalizi (2001) précise le concept d'émergence intrinsèque comme étant la simple caractérisation d'un processus créant un autre processus avec des capacités améliorées. Bersini (2004) défend l'émergence intrinsèque pour la relaxation apportée de la subjectivité usuellement présente dans la définition de l'émergence. Il l'interprète alors comme une machine remplaçant l'observateur humain. Cette machine adapte ses observations aux phénomènes macroscopiques produits par le système et réintroduit l'information obtenue dans le système. En somme, le système devient capable d'introspection. Il est capable d'observer ses propres propriétés abstraites et de les assimiler (Philemotte and Bersini, 2007).

Une Gestalt de représentation résulte en une modification intrinsèque des opérateurs, et donc de tout le processus de la métaheuristique. Un nouveau processus en dérive en conséquence. La Gestalt permet d'extraire les caractéristiques les plus significatives pour la métaheuristique afin de l'assister. Considérant le nouveau processus de recherche expérimenté, la Gestalt de représentation peut être adaptée. L'heuristique de la Gestalt enrichit la métaheuristique d'une propriété d'émergence intrinsèque.

Nous pouvons donner une perspective beaucoup plus générale de toute cette dynamique. Compte tenu du point de vue présenté dans la Sec. 3.2.1, l'évaluation des actions d'une métaheuristique dans un environnement, l'espace de recherche, permet de construire de nouvelles lentilles. Ces lentilles changent la perception de la métaheuristique et conséquemment l'environnement. Et ce changement altère les actions menées par la métaheuristique et ainsi de suite. L'heuristique de la Gestlat crée une rétroaction causant un changement du comportement du système. Nous avons ici les conditions cybernétiques minimales reconnues des systèmes capables de régulation et d'adaptation.

6.7 Conclusion

Une Gestalt description ne correspond pas à un schema et les Gestalt variables ne sont pas des blocs constitutifs. L'heuristique de la Gestalt n'est ni une méthode de recherche à base de modèle, ni un type d'algorithme de raffinage multi-niveaux, ni un AE d'apprentissage de lien, ni un AE compositionnel. Par contre, elle partage avec la recherche à voisinage variable un même type de mécanisme d'adaptation de voisinage, mais ce mécanisme n'est pas conduit selon les mêmes objectifs. Et elle peut être considérée

comme une hyperheuristique particulière. Finalement, l'heuristique de la Gestalt enrichit la métaheuristique assistée d'une propriété d'émergence intrinsèque.

7

Conclusion

Dans le domaine de l'optimisation combinatoire, il est une méthode algorithmique approximative qui s'est démarquée ces dernières années de par sa malléabilité, sa généralité et son efficacité : la métaheuristique. Bien qu'ayant pour objectif d'être détachées d'un problème particulier, elle nécessite certaines attentions afin de se positionner comme un fleuron de l'état de l'art. Aussi, compte tenu d'un problème donné, il convient de soigneusement choisir les opérateurs, les paramètres de configuration et la représentation. L'efficacité n'est donc qu'au prix d'un temps supplémentaire de développement.

Cette spécialisation n'est bien entendu pas toujours possible parce que le problème est que trop peu compris ou mal défini, on n'a pas à disposition la spécialisation ad hoc, les ressources nécessaires au développement sont insuffisantes, ou tout simplement le souhait d'effectuer un prototypage rapide. Ces cas de figures s'inscrivent dans le scénario de la boîte noire : le problème est vu comme une boîte dont rien ne transpire si ce n'est l'évaluation d'une possible solution. Vient alors la question légitime : pouvons-nous y remédier ?

Oui, nous pouvons, notamment en poursuivant l'objectif suivant, qui a fait le sujet de cette thèse :

> *Elaborer une méta-modélisation dynamique inspirée de la psychologie de la Gestalt permettant d'assister en ligne une métaheuristique dans sa tâche d'optimisation et ce dans les conditions proches du scénario de la boîte noire.*

Par assistance, nous entendons une amélioration de la qualité de la solution à temps constant ou d'une diminution du temps nécessaire à qualité constante de solution. Mais que sont-ce que la méta-modélisation et la psychologie de la Gestalt ?

La psychologie de la Gestalt fournit une théorie générale tentant de comprendre certains processus mentaux tels que la vision ou la résolution de problèmes comme le produit d'une systémique. Selon cette théorie, une totalité ne peut se résumer à la simple somme de ses parties. Et les parties sont plus que son individualité compte tenu du contexte. La structuration de la totalité et des parties est alors expliquée par des lois et des principes ad hoc. Inspiré de cette théorie descriptive, un concept analogue peut être développé dans le cadre de l'optimisation combinatoire et des métaheuristiques. Pour ce faire, un parallèle est établi entre les humains résolvant un problème et les métaheuristiques traitant une instance de POC. Ce parallélisme nous amène à considérer la représentation comme la perception qu'a la métaheuristique du problème. Ainsi, changer la représentation modifie la perception de la métaheuristique et donc sa manière d'agir, puisque le graphe de voisinage résultant de la représentation est modifié en conséquence. Et la méta-modélisation dans tout cela ?

La méta-modélisation est une approche plus générale dans laquelle s'inscrit ce concept de la Gestalt. Cette approche a pour objectif d'assister en ligne une métaheuristique dans sa tâche d'optimisation. Une Gestalt de représentation peut en effet être vue comme un méta-modèle de recherche, autrement dit un cadre permettant de construire et/ou modifier le modèle de recherche qui n'est autre que le graphe de voisinage. La construction d'une Gestalt de représentation est dès lors une méta-modélisation de recherche capable d'assister en ligne une métaheuristique dans sa tâche d'optimisation.

Voyons en quoi consiste la Gestalt appliquée aux métaheuristiques. Une Gestalt canonique de la représentation d'une instance de POC revient alors à l'agrégation de variables en de nouvelles entités constituantes pouvant être manipulées par les opérateurs en tant que telles. Chaque opérateur est de ce fait généralisé par la délégation de son opération aux parties de l'ensemble qu'il manipule. Ainsi, non seulement une Gestalt de représentation est une abstraction structurelle de par l'agrégation de variables, mais elle est également une abstraction fonctionnelle du processus de recherche. La construction d'une Gestalt s'articule, pour sa part, autour de trois principes inspirés de la psychologie de la Gestalt : l'investigation, la perturbation et la contextualisation. L'investigation a pour objectif de comprendre la structure de l'instance de POC traité et la production en résultat de cet apprentissage de nouvelles Gestalts. Pour ce faire, elle se base sur les principes de perturbation et de contextualisation. La perturbation assure l'innova-

tion et l'adaptation en introduisant explicitement de petites variations ou erreurs. Et la contextualisation met en pratique une Gestalt et évalue son utilité. Cette évaluation dirige les règles génériques de construction mises en oeuvre par l'investigation.

Ce concept, que nous nommons l'heuristique de la Gestalt, peut se formaliser comme suit. Une Gestalt de représentation reformule une instance de POC selon une Gestalt description G définie comme une partition de l'ensemble des variables X, contenant ainsi des sous-ensembles g de variables appelées Gestalt variables. En conséquence, les solutions candidates sont reformulées selon une nouvelle représentation $\gamma : S \to \mathcal{V}_\gamma$ faisant correspondre toute solution candidate $s \in S$ à un m-uplet de valeur $c_\gamma(g) \in \mathbb{D}_g = \prod_{x \in g} \mathbb{D}_x$ pour chaque Gestalt variable $g \in G$. Ce changement de représentation a pour effet de modifier les opérateurs appliqués. Ce changement s'incarne dans la lentille associée qui est un opérateur mathématique $\mathcal{G}$ appliquant la fonction de voisinage n, résultant de l'opérateur et de la représentation initiale, sur une nouvelle fonction de voisinage n_γ. Il en résulte un nouveau graphe de voisinage v_γ dont le degré de voisinage pourrait être diminué. De ce fait, la lentille assure la généralisation de l'opérateur aux Gestalt variables par sa délégation. L'heuristique de la Gestalt est, quant à elle, responsable de la construction des nouvelles lentilles. Elle peut être définie comme la fonction $B(\mathcal{G}, C) = \mathcal{G}'$ qui maximise la qualité d'assistance donnée par le critère C. Après application d'une lentille $\mathcal{G}$ et son évaluation selon C, la contextualisation est réalisée. L'investigation et la perturbation sont quant à elles réalisées au sein de la fonction B.

Reste à savoir comment l'heuristique de la Gestalt peut-elle être appliquée à une métaheuristique et comment peut-elle être implémentée. Nous répondons à cette question en ne considérant qu'une famille de métaheuristique, les AEs. L'assistance d'un AE par l'heuristique de la Gestalt n'en nécessite pas la modification, mais plutôt l'extension de son implémentation. Cette extension permettra la transformation de la représentation originale en sa version Gestalt et la généralisation des opérateurs aux Gestalt variables par la délégation aux variables composantes.

Considérant que l'AE assisté est un SAG, nous fournissons une implémentation évolutionniste de l'heuristique de la Gestalt. Celle-ci repose sur l'utilisation d'un AGG pour évoluer les lentilles. Ces dernières, ou plutôt leurs Gestalt descriptions associées, sont alors représentées au moyen d'une représentation de groupement. La population du SAG se voit ensuite structurée en sous-populations. Sur chacune d'elles, il est appliqué une lentille. L'évaluation d'une lentille se fait alors sur base de la qualité des solutions trouvées au sein de sa sous-population associée. La sélection des lentilles implique celle de sa sous-population associée. De ce fait, les solutions candidates se retrouvent être sélectionnées selon plusieurs niveaux. Finalement,

lors de la recombinaison de deux lentilles, une migration de solutions candidates est effectuée entre leurs sous-populations associées.

Quatre études expérimentales ont été menées afin d'éprouver séparément les concepts sous-jacents de l'heuristique de la Gestalt, d'étudier et de valider son implémentation que nous venons d'exposer ci-avant, et de sa mise en situation réaliste. Des résultats obtenus, nous avons pu soutenir un certain nombre d'assertions (reprises aux Chapitres 1 et 5) dont nous avons tiré les interprétations suivantes :

- l'heuristique de la Gestalt peut se constituer comme une excellente méthode d'initialisation de population de solutions,
- l'heuristique de la Gestalt s'avère un moyen générique et simple d'inoculer de la connaissance à propos du problème dans un algorithme génétique,
- l'heuristique de la Gestalt se constitue comme une bonne première méta-modélisation dynamique et en ligne pour l'assistance des algorithmes génétiques dans le cadre d'un scénario de la boîte noire, et
- l'AGG ayant démontré ses limitations, la révision de l'implémentation de l'heuristique de la Gestalt pourrait améliorer les performances actuellement obtenues.

En conclusion, l'heuristique de la Gestalt s'avère une approche prometteuse pour assister, à tout le moins les AGs compte tenu d'un scénario de la boîte noire. L'implémentation réalisée arbore des comportements espérés au niveau conceptuel, mais montre un certain nombre de limites. Et de nombreuses autres implémentations et extensions restent possibles. Sans compter son application à d'autres métaheuristiques ou encore l'évolution de certains concepts clefs comme les Gestalt description. C'est pourquoi pour terminer la dissertation, nous présentons un certain nombre de travaux futurs de cet acabit.

7.1 Assister d'autres métaheuristiques

Nous avons défini l'heuristique de la Gestalt indépendamment de toute considération de réalisation. Formellement, elle peut assister n'importe quelle métaheuristique, puisque ces dernières partagent de mêmes fondations, telles que le graphe de voisinage ou la représentation. Toutefois, seule l'assistance d'AEs a été détaillée et testée expérimentalement. Nous ne pouvons donc pas nous avancer sur la réussite effective de l'assistance d'un autre type de métaheuristique. S'il est souhaité de démontrer l'intérêt général de l'heuristique de la Gestalt, il conviendra de la tester sur plusieurs autres métaheuristiques. Cette mise à l'épreuve permettra également d'a-

guerrir et de raffiner l'implémentation de l'heuristique de la Gestalt, voire de la réformer totalement, comme nous l'exposons dans la section suivante.

7.2 Implémentation de l'heuristique de la Gestalt

L'heuristique de la Gestalt pose principalement un problème d'apprentissage en ligne des lentilles. Ce problème peut bien entendu être formulé comme un problème d'optimisation. L'implémentation de l'heuristique pourrait donc aussi bien faire appel à des techniques d'apprentissage que d'optimisation. L'heuristique est en effet comparable à une interface d'hybridation permettant d'additionner un second algorithme d'apprentissage ou d'optimisation à un premier algorithme de la famille des métaheuristiques. De nombreuses combinaisons sont alors possibles.

S'il convenait de choisir un algorithme d'optimisation pour réaliser une nouvelle implémentation de l'heuristique de la Gestalt, un choix particulièrement judicieux, pour remplacer celui actuellement porté sur les AEs, serait d'utiliser les algorithmes de recherche à base de modèle (voir Sec. 6.1). Rappelons que l'heuristique de la Gestalt s'avère être une sorte de métamodélisation, comme nous l'avons discuté à la Sec. 3.3. Il s'agit donc bien de construire un modèle grossier du processus de recherche de la métaheuristique assistée. Avec un algorithme de recherche à base de modèle, il conviendrait de construire explicitement le modèle probabiliste du processus de recherche. Prenons, par exemple, l'algorithme d'optimisation par colonie de fourmis. Il s'agirait d'utiliser des fourmis pour suivre les actions de la métaheuristique et déposer des phéromones sur le sous-graphe correspondant de l'espace de recherche. Ce sous-graphe permettrait de produire le graphe de relations entre variables et par là-même les lentilles correspondantes.

Une alternative, comme énoncée plus haut, serait l'utilisation de techniques d'apprentissage telles que celles par renforcement développées spécifiquement pour le routage stochastique d'agents sur graphe. Dans la Sec. 2.2.3, il a été rappelé que le processus de recherche d'une métaheuristique peut être considéré comme une marche sur le graphe de voisinage. Pour les métaheuristiques basées sur le parcours, le processus est similaire à un agent qui marche sur le graphe de voisinage ν. Pour celles basées sur la population de solutions candidates, il ne s'agit plus d'un seul agent, mais d'une population d'agents pouvant éventuellement communiquer ensemble. Dans les deux cas, un ou plusieurs agents marchent sur le graphe de voisinage et recherchent l'optimum global, l'objectif, avec la contrainte de ne pouvoir le reconnaître parfaitement. Chaque pas d'agent est dirigé par les opérateurs et coûte une même quantité de temps, une unité temporelle

d'itération. Suite à chaque pas, un agent peut être positivement ou négativement récompensé par la récompense $r_{ss'}$. Elle dépend de la progression de l'agent vers l'objectif. Ce scénario, tel que nous venons de le décrire, n'est autre que le routage stochastique d'agents (Achbany et al., 2006). Selon cette interprétation, la stratégie de routage n'est rien de plus que les opérateurs paramétrés que la métaheuristique utilise, la stratégie idéale étant l'ensemble des probabilités de transition $\{p_{ss'}\}$ qui maximise la récompense totale, une mesure de la proximité de l'objectif. Cette perspective fournirait une intéressante implémentation de l'heuristique de la Gestalt, avec la difficulté que le graphe de voisinage ne peut être connu que partiellement compte tenu de sa taille. Les lentilles seraient alors produites sur base de la matrice de probabilités de transition.

Mais l'implémentation n'est pas la seule piste possible d'évolution. Le concept même de la Gestalt pourrait être étendu, comme nous allons le proposé dans la section suivante.

7.3 Une Gestalt plus sophistiquée

Une Gestalt variable, telle que définie à la Sec. 4.1.1, est un ensemble de variables $G := \{\ldots, y_j, \ldots\}$. Dans cet ensemble G, aucune relation n'est définie entre les variables. De même, dans une Gestalt description Γ, les Gestalt variables ne sont pas mises en relation. Partant de ces limitations, il est possible d'imaginer un certain nombre d'extensions du concept actuel de la Gestalt :

- la constitution d'une Gestalt variable sur base d'autres Gestalt variables, permettant ainsi une composition hiérarchique,
- l'établissement d'un lien entre les valuations des variables au sein d'une même Gestalt variable, ou
- l'établissement d'un lien entre les valuations des Gestalt variables au sein d'une même Gestalt description.

Ces suggestions renforcent le principe de composition fonctionnelle de la Gestalt et nécessitent une grammaire pour être facilement décrites. L'heuristique de la Gestalt donnera alors lieu à une réelle reprogrammation de la représentation. Ceci nécessitera des techniques spéciales telles que la programmation génétique (Poli et al., 2008) ou l'évolution grammaticale (Ryan et al., 1998).

A SAG

L'AG standard traitant des problèmes à variables binaires est le SAG (Goldberg, 1989; Mitchell, 1996). Son efficacité repose principalement sur l'opérateur de recombinaison, la mutation assurant principalement la diversification. Voyons plus en détail son fonctionnement tel qu'il est présenté à l'Alg. 4. Dans un premier temps, une population de N_{pop} individus est créées et évaluées. Le SAG entame alors la répétition de la génération d'une nouvelle population en recombinant les "meilleurs" individus sélectionnés.

A.1 Population et individu

Le SAG utilise une population de taille constante N_{pop} sans recouvrement. La caractéristique de non recouvrement exprime qu'à chaque génération, la population est totalement renouvelée

Le chromosome d'un individu consiste simplement en un vecteur de bits $\bar{x} = (x_1, \ldots, x_n)$ de taille fixe n où chaque gène $x_i \in \{0,1\}$. Un exemple de chromosome est $(0,1,1,1,0,0)$.

A.2 Initialisation

L'initialisation de la population est la première étape accomplie par le SAG. Les N_{pop} individus sont alors générés aléatoirement et évalués. La génération aléatoire obéit généralement à une loi de probabilité uniforme sur l'espace de recherche $\{0,1\}^n$. En pratique, les gènes d'un chromosome

Algorithm 4 Simple Algorithme Génétique (SAG)

1: **initialisation** d'une population de N_{pop} individus
2: **while** critère d'arrêt non satisfait **do**
3: **sélection** de N_{pop} individus pour reproduction
4: **recombinaison** et production de N_{pop} descendants
5: **mutation** des N_{pop} descendants
6: **évaluation** de la nouvelle génération
7: **replacement** de l'ancienne population par la nouvelle
8: **end while**

sont générés selon une distribution de Bernoulli avec une probabilité de réussite non biaisée d'un demi.

A.3 Sélection

La sélection des meilleurs individus pour la reproduction s'effectue proportionnellement aux niveaux d'adaptation des individus (Holland, 1975). La probabilité de sélection d'un individu s'obtient alors par

$$\frac{\hat{f}(x(i))}{\sum_{j=1}^{N_{pop}} \hat{f}(x(j))} .$$

En corollaire, la probabilité de sélection d'un individu augmente en fonction de son niveau d'adaptation.

Généralement, la sélection est implémentée comme une "roulette". Chaque individu est alors assigné à une case dont l'aire est proportionnelle à son niveau d'adaptation (Mitchell, 1996). Cette implémentation possède un désavantage majeur dû à l'échantillonnage répété : le nombre effectif de descendants d'un individu peut s'écarter fortement du nombre attendu. Baker (1987) a suggéré une solution à ce biais. Le EUS utilise un seule valeur aléatoire pour échantillonner tous les individus en les choisissant à intervalles régulièrement espacés (Baker, 1987). La roulette n'est alors tournée qu'une seule fois pour générer les N_{pop} descendants à partir des N_{pop} parents assurant ainsi le respect de la loi proportionnelle.

A.4 Recombinaison

Le SAG utilise une Recombinaison à Un Point (R1P) pour accoupler les individus sélectionnés (Holland, 1975). L'R1P consiste à choisir aléatoirement un point de section dans l'ensemble des positions des gènes $\{1,\ldots,n-1\}$, les positions étant numérotées de 1 à n. Deux descendants

sont alors générés en échangeant, entre les parents, les portions de chromosome de part et d'autre du point de section. Prenons, par exemple, les deux parents $(0,1,1,1,0,0)$ et $(0,0,0,1,1,1)$. Et le point de section est 3. Les descendants obtenus sont donc $(0,0,0|1,0,0)$ et $(0,1,1|1,1,1)$. Pour terminer, ajoutons que la recombinaison est appliquée avec une probabilité P_X, la non application revenant alors à une simple recopie des parents.

A.5 Mutation

Le SAG applique une mutation à basculement sur tous les individus générés suite à la recombinaison. Cette mutation consiste en le basculement de chaque bit, gène du chromosome avec une probabilité P_μ.

B

AGG

L'AGG, introduit par Falkenauer (1994), consiste en une extension des AGs capables de traiter des problèmes de groupement. Cette extension consiste principalement en une représentation génétique apte à représenter directement la notion de groupement, et des opérateurs génétiques (mutation, inversion et recombinaison) pouvant la manipuler sans atteinte à l'intégrité des groupements représentés (Falkenauer, 1998). Pour ce qui est de l'opérateur de sélection, Falkenauer (1998), il se résume à une sélection bruitée par tournoi. Bien entendu, il est possible d'en utiliser d'autres types. Un résumé de l'AGG et des ses principales étapes est présenté à l'Algorithme 5.

Algorithm 5 Algorithme Génétique de Groupement (AGG)

1: **initialisation** et **évaluation** de la population
2: **while** critère d'arrêt non satisfait **do**
3: **sélection** de la population de reproducteurs
4: **copie** des élites dans la nouvelle génération
5: **recombinaison** des reproducteur et production de la descendance
6: application de la **mutation** et l'**inversion** à la descendance
7: **évaluation** de la nouvelle génération
8: **end while**

B.1 Représentation

Originairement, la représentation de groupement proposée par Falkenauer (1994) est composée de deux parties : une première décrivant la ré-

partition des objets entre les différents groupes, et une seconde définissant l'appellation des groupes utilisés. Cette dernière partie, , bien que quelque peu redondante, est requise par la recombinaison de l'AGG, comme nous le verrons dans la section suivante.

La première partie, la partie des objets, est un vecteur de n valeurs d'étiquette de groupe, où n est le nombre d'objets. La seconde partie, la partie des groupes, est une liste de taille variable de m étiquettes dénommant chacune un groupe. Selon sa position ordinale, autrement dit son locus, le gène détermine l'appartenance d'un objet en particulier au groupe correspondant à l'étiquette donnée par l'allèle. Pour exemple, l'individu encodé par

$$\underbrace{BCCAB}_{\text{objets}} : \underbrace{CAB}_{\text{groupes}}$$

représente le groupement de cinq objets en trois groupes A, B et C. Les premier et dernier objets appartiennent au groupe B ; le pénultième objet au groupe A ; et les second et troisième objets au groupe C.

B.2 Recombinaison

La recombinaison pour le groupement, opérateur complexe appliqué avec une probabilité P_X, nécessite plusieurs étapes successives pour être accomplie, l'idée principale étant d'assurer l'échange du matériel génétique pertinent, autrement dit la définition des groupements. Son algorithme [1] peut être décrit comme suit :

1. Sélection aléatoire de deux paires de positions dans la partie des groupes des parents 1 et 2, positions délimitant la section de recombinaison C_1 et C_2, c'est-a-dire une sous-liste d'étiquettes de groupe.
2. Echange des sections de recombinaison C_1 et C_2 et création des enfants 1 et 2.
3. Dans la partie des objets de l'enfant 1 (respectivement 2), injection des éléments de la partie des objets du parent 2 (resp. 1) pour lesquels les valeurs correspondent aux étiquettes reprises dans C_2 (resp. C_1).
4. Pour les objets de l'enfant 1 (resp. 2) n'ayant pas encore reçus une valeur d'appartenance, injection des éléments de la partie des objets du parent 1 (resp. 2) pour lesquels les valeurs correspondent aux étiquettes reprises dans la partie des groupes du parent 1 (resp. 2) excepté ceux de C_1 (resp. C_2), et sans écrasement des valeurs déjà injectées dans l'étape précédente.

1. Pour sa définition formelle, le lecteur est invité à consulter Tucker et al. (2005).

5. Définition aléatoire de l'appartenance des objets des deux enfants, n'ayant pas encore été définie au cours des deux étapes précédentes, selon leur partie de groupes respective.
6. Suppression, dans la partie des groupes des deux enfants, des étiquettes de groupes restés vides.

Considérons par exemple les deux parents $bccab : cab$ et $ABBAA : BA$, et effectuons une possible recombinaison :

Etape 1 : Sélection des sections de recombinaison : $c|a|b$ et $|B|A$.

Etape 2 : Création de la partie des groupes des enfants : cBb et aA.

Etape 3 : $xBBxx : cBb$ et $xxxax : aA$, où x dénote une étiquette indéfinie.

Etape 4 : $bBBxb : cBb$ et $xxxax : aA$.

Etape 5 : $bBBbb : cBb$ et $AaaaA : aA$.

Etape 6 : $bBBbb : Bb$ et $AaaaA : aA$.

B.3 Mutation

Trois sortes de mutation sont définies et utilisées :
– le déplacement d'un objet d'un groupe vers un autre,
– la division d'un groupe en deux, et
– la fusion de deux groupes en un seul.

B.3.1 Redéfinition de l'appartenance d'un objet

Avec une probabilité $P_{R\mu}$, chaque objet peut être déplacé vers un autre groupe existant, choisi aléatoirement. Cette mutation opère au sein de la partie des objets sans affecter la partie des groupes. Aucun groupe n'est donc créé ou supprimé.

Considérons l'individu $bccab : cab$, une possible application de cette mutation serait le déplacement du quatrième objet vers le groupe b et nous donnerait alors :

$$bccbb : cb\ .$$

B.3.2 Division d'un groupe

Avec une probabilité $P_{S\mu}$, un groupe est aléatoirement choisi et divisé en deux nouveaux groupes. Pratiquement, seule une nouvelle étiquette de groupe est ajoutée dans la partie correspondante, et les objets du groupe divisé sont déplacés aléatoirement dans ce nouveau groupe. Cette mutation

opère au sein de la partie des groupes et crée un nouveau groupe. A remarquer qu'elle ne peut être utilisée s'il n'existe que des groupes composés d'un seul objet.

Considérons l'individu *bccab* : *cab*, une possible application de cette mutation serait la division le groupe *c* de la manière suivante :

$$bcdab : cabd \ .$$

B.4 Fusion de deux groupes

Avec une probabilité $P_{M\mu}$, deux groupes distincs sont aléatoirement sélectionnés et fusionnés en un seule et unique nouveau groupe. En pratique, l'étiquette d'un des deux groupes est retirée de la partie des groupes et les objets du groupe correspondant sont déplacés vers l'autre restant. Cette mutation opère au sein de la partie des groupes et supprime un groupe existant. A remarquer qu'elle ne peut être appliquée s'il n'existe qu'un seul groupe.

Considérons l'individu *bccab* : *cab*, une poosible application de cette mutation serait la fusion des groupes *a* et *b* :

$$accaa : ca \ .$$

B.5 Inversion

Avec une probabilité P_I, l'inversion s'opère au sein de la partie des groupes. Deux positions, dans la liste, sont tirées aléatoirement. Entre eux, l'ordre des étiquettes de groupe est simplement inversé. Cet opérateur ne change pas la composition des groupes et permet d'éviter le biais de sélection des points de section de recombinaison.

Considérons l'individu *bcdab* : *cabd*, une possible inversion serait conditionnée par la sélection de deux positions, telles que

$$bcdab : c|ab|d \ ,$$

et l'inversion de l'ordre entre ces deux positions :

$$bcdab : cbad \ .$$

C

AGO

L'AGO est un AG, introduit par Goldberg (1989), pour pouvoir traiter des problèmes d'ordonnancement, de classement, d'agencement, ou d'arrangement, dont le PVC est un exemple. La représentation utilisée par l'AGO se base sur la permutation d'objets. Ainsi, le PVC est-il simplement et naturellement encodé sous la forme d'un chemin (Goldberg, 1989; Radcliffe and Surry, 1994). Dans la suite, nous détaillerons d'ailleurs l'AGO en l'illustrant par son application au PVC.

C.1 Représentation

Comme déjà mentionné, une représentation naturelle pour le PVC est la permutation d'étiquettes de ville, décrivant par là-même un chemin passant par chacune des villes (Goldberg, 1989; Radcliffe and Surry, 1994). La position des étiquettes dans la permutation détermine l'ordre dans lequel la ville correspondante sera visitée. Par exemple, si nous considérons quatre villes dénommées de 1 à 4, un possible tour est (4,2,1,3) où la ville 4 est la première ville à être visitée, et la ville 3 la dernière. Au vu du caractère cyclique du PVC, la permutation (2,1,3,4) représente bien entendu le même tour. De même, à cause de la symétrie du PVC, la permutation (1,2,4,3) représente le même tour. Pour une même solution candidate, nous avons donc $2n$ représentations possibles. Pour réduire cette redondance à 2 représentations, il suffit de fixer la première ville arbitrairement.

Si nous reprenons la formalisation présentée à la Sec. 2.1.2.3 du Chapitre 2, nous pouvons dès lors décrire notre exemple sur base de l'ensemble $X =$

{1,2,3,4} avec pour domaine $\mathbb{D}$ commun associé l'ensemble $X \setminus \{1\}$. En effet, une valuation consiste en une permutation σ sur l'ensemble $X \setminus \{1\}$. Par exemple (1,2,4,3), où la position dans le quadruplet définit la permutation $\sigma(2) = 2$, $\sigma(4) = 3$, et $\sigma(3) = 4$. La bijection φ associera donc chaque solution candidate à deux permutations. Nous avons une représentation dite dégénérée, autrement dit plusieurs représentations représentent la même solution candidate (Radcliffe and Surry, 1994).

C.2 Recombinaison

De nombreux opérateurs de recombinaison ont été conçus pour résoudre les POC dont les solutions peuvent être représentées sous la forme de permutations. Ces opérateurs peuvent être classés du plus générique, tel que la RCP (Goldberg, 1989), au plus spécifique tel que le CRA (Whitley et al., 1989). Dans tous les cas, ils sont appliqués avec une probabilité P_X. Abordons plus en détail la RCP et le CRA.

C.2.1 RCP

La RCP est un opérateur hybride de recombinaison transmettant à la fois l'ordre relatif, la position absolue et l'adjacence présents dans les permutations représentant les individus parents Moraglio (2007). Sous la RCP, deux permutations sont alignées et deux points de section sont tirés aléatoirement. Ces points définissent une section de correspondance bijective d'échange entre certaines étiquettes de ville des parents.

Considérons, par exemple, les deux parents (1,4,2,3) et (1,4,3,2), et les points de sections 1 et 3 :

$$(1|4,2|3)$$
$$(1|4,3|2)\ .$$

La section ainsi définie nous donne les correspondances suivantes :

$$4 \leftrightarrow 4$$
$$2 \leftrightarrow 3\ .$$

Ces étiquettes sont alors échangées entre les parents au sein de la section :

$$(1|4,3|3)$$
$$(1|4,2|2)\ .$$

Finalement, les étiquettes hors de la section sont remplacées selon les correspondances définies plus tôt :

$$(1|4,3|2)$$
$$(1|4,2|3)\ .$$

C.2.2 CRA

Le CRA est un croisement expressément développé pour le PVC : il exploite l'adjacence, la plus importante propriété du PVC (Moraglio, 2007). En considérant la solution candidate en termes d'arêtes plutôt que de sommets, c'est-à-dire les villes, il recrée un chemin similaire à l'ensemble des chemins existants définis par les parents. Ce faisant, il introduit un nombre minimum d'arêtes étrangères aux parents Whitley et al. (1989).

Pour exemple, considérons les deux parents (1,2,3,4,5) et (1,2,3,5,4). Partant d'eux, la carte suivante d'arêtes, qui liste les voisins de chaque ville dans chaque parent, peut être construite :

$$1:\ 2,4,5$$
$$2:\ 1,3$$
$$3:\ 2,4,5$$
$$4:\ 3,5,1$$
$$5:\ 1,4,3\ .$$

Afin de générer un descendant, le CRA commence par sélectionner aléatoirement une ville de départ et retire toutes ses occurrences de la carte d'arêtes. Ensuite, le CRA choisit, parmi ses voisins, la ville avec le moins de voisins. Si plusieurs choix restent possibles, ce dernier est discriminé aléatoirement. Les occurrences de cette ville sont alors retirées de la carte des arêtes. S'il n'y a plus de voisins, la prochaine ville est simplement sélectionnée aléatoirement dans la carte d'arêtes. Le CRA renouvelle cette procédure jusqu'à la transmission de toutes les villes à l'enfant. Ainsi, un possible descendant pourrait être construit comme suit :

$$(1)$$
$$(1,2)$$
$$(1,2,3)$$
$$(1,2,3,5)$$
$$(1,2,3,5,4)\ .$$

C.3 Mutation par échange de loci adjacents

Cette mutation sélectionne aléatoirement un seul locus et l'un de ses deux voisins, et échange les allèles des gènes s'y trouvant. Elle est appliquée avec une probabilité P_μ sur chaque enfant produit après recombinaison. Prenons, par exemple, l'individu $(2,1,4,3)$. La paire $(4,1)$ est sélectionnée et échangée pour donner la permutation $(3,1,4,2)$.

C.4 Sélection par tournoi

Ce type de sélection (Goldberg et al., 1989) consiste à organiser un tournoi entre un certain nombre d'individus sélectionnés aléatoirement. Le tournoi est gagné par l'individu ayant le meilleur niveau d'adaptation, lui donnant ainsi le droit de se reproduire avec le vainqueur d'un autre tournoi. Les vainqueurs peuvent être retirés ou non de la population.

D
Modèle Orienté Objet de la Gestalt

L'intégration de l'heuristique de la Gestalt au sein d'une métaheuristique donnée, et dans notre un AG, nécessite un certain nombre de modifications de la structure de données permettant de modéliser le concept de solution candidate. Nous avons résolu ce problème dans le respect du paradigme de la programmation orientée objet.

La Figure D.1 présente le diagramme de classe permettant de modéliser le concept d'une solution candidate représentée selon une Gestalt description. Passons tout d'abord en revue la partie permettant de représenter une solution candidate normalement.

Le concept de solution candidate est modélisé par la classe **Solution**. Une **Solution** ne peut être représentée sans une description, c'est-à-dire sa définition en termes de variables. Aussi, une **Solution** est associée à une et une seule **Description**. La **Description**, comme nous venons de le dire, permet de définir le moyen de représentation d'une **Solution** : des variables associées à un certain domaine de valeur. La **Description** est donc composée d'au moins une **Variable**. Cette composition s'exprime sous la forme d'une autre classe définissant le type de la variable, autrement dit son domaine de valeurs : la classe **Domain**. La représentation d'une solution étant dès lors définie, la Solution est alors représentée réellement par son agrégation de valuation des **Variable**s définie par **Description**. La valuation est modélisée par une classe d'association, la classe **Value** qui compose la classe **Domain**, l'ensemble des **Value**s possibles que peut prendre la **Variable** correspondante.

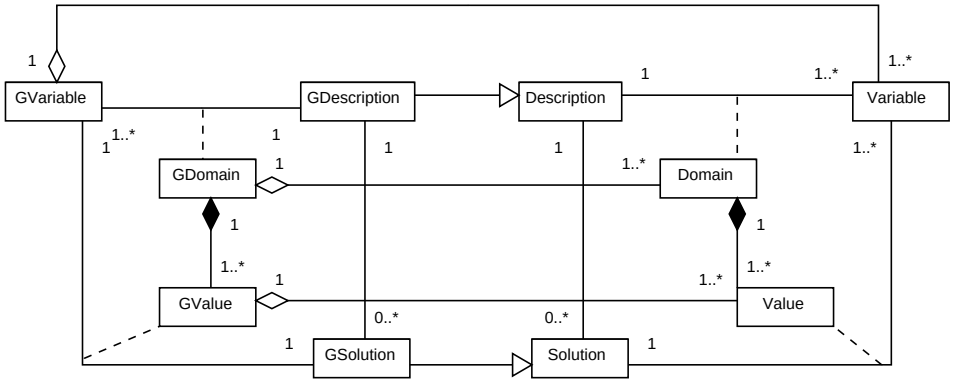


FIGURE D.1 – Le diagramme de classe reprend synthétiquement le principe de la modélisation d'un solution candidate représentée selon une Gestalt description.

Pour pouvoir représenter une solution candidate selon une Gestalt description, tout en assurant toujours la description initiale de telle sorte que la transformation ne soit pas coûtante, en particulier lors de chaque évaluation, nous considérons un concept dérivé de **Solution** : **GSolution**. Une **GSolution** est une **Solution** représentée selon sa **Description** originale et une Gestalt description. La Gestalt description est modélisée par la classe **GDescription**, une classe dérivée de **Description**. Une **GDescription** définit le mode de représentation en termes de **GVariables** auxquelles sont associés des **GDomain**. La classe **GVariable** représente le concept de Gestalt variable, autrement dit une agrégation de **Variable**s. A chaque **GVariable**, est associé un **GDomain**, son domaine associé construit sur les **Domain**s des **Variable**s qui la composent. La **GDescription** définira donc non seulement la représentation originale puisqu'il hérite de la définition de sa classe parente en termes de **Variable**s, mais également la Gestalt de représentation en termes de **GVariables**. Dès lors, une **GSolution** est réalisée par les valuations des **GVariables**, représentée par la classe d'association **GValue**. Celle-ci compose le **GDomain** et se définit sur base des **Values** correspondant aux **Domains** agrégés. C'est dans **GValue** que l'on généralise l'application d'un opérateur sur les différentes **Value**s, par la délégation de cette opération sur les **Value**s la composant.

Bibliographie

Youssef Achbany, Francois Fouss, Luh Yen, Alain Pirotte, and Marco Saerens. Tuning continual exploration in reinforcement learning. In S. Kollias, A. Stafylopatis, W. Duch, and E. Oja, editors, *Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 06)*, volume 4132 of *Lecture notes in Computer Science*. Springer, September 2006.

Mitchell Ash. *Gestalt Psychology In German Culture 1890 - 1967*. Cambridge University Press, 1995.

James E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 14–21, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.

Shumeet Baluja. Population-based incremental learning : A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Computer Science Department, Carnegie Mellon University, 1994.

Laura Barbulescu, Jean-Paul Watson, and Darrell L. Whitley. Dynamic representations and escaping local optima : Improving genetic algorithms and local search. In *AAAI/IAAI*, pages 879–884, 2000.

Thomas Bartz-Beielstein. *Experimental Research in Evolutionary Computation : The New Experimentalism (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

Roberto Battiti, Mauro Brunato, and Franco Mascia. *Reactive Search and Intelligent Optimization*. Operations research/Computer Science Interfaces. Springer Verlag, 2008. in press.

Hugues Bersini. Whatever emerges should be intrinsically useful. In *Artificial life 9*, pages 226–231. The MIT Press, 2004.

Mourad Bessaou, Alain Pétrowski, and Patrick Siarry. Island model cooperating with speciation for multimodal optimization. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI*, pages 437–446, Berlin, 2000. Springer.

Mauro Birattari. *The Problem of Tuning Metaheuristics as Seen from a Machine Learning Perspective*. PhD thesis, Université Libre de Bruxelles, Brussels, Belgium, 2004.

Mauro Birattari, Thomas Stützle, Luis Paquete, and Klaus Varrentrapp. A racing algorithm for configuring metaheuristics. In *GECCO '02 : Proceedings of the Genetic and Evolutionary Computation Conference*, pages 11–18, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization : Overview and conceptual comparison. *ACM Computing Survey*, 35 (3) :268–308, 2003.

Yossi Borenstein and Riccardo Poli. Structure and metaheuristics. In *GECCO '06 : Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1087–1094, New York, NY, USA, 2006. ACM.

Vicki Bruce, Patrick R. Green, and Mark Georgeson. *Visual Perception : Physiology, Psychology and Ecology*. Psychology Press Ltd, 3d edition, 1996.

Edmund Burke, Graham Kendall, Jim Newall, Emma Hart, Peter Ross, and Sonia Schulenburg. Hyper-heuristics : An emerging direction in modern search technology. pages 457–474. 2003.

Thomas Bäck. *Evolutionary Algorithms in Theory and Practice : Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.

Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors. *Handbook of Evolutionary Computation (Computational Intelligence Library)*. Oxford University Press, ringbound edition, April 1997.

Jiah-Shing Chen, Yao-Tang Lin, and Liang-Yu Chen. A relation-based genetic algorithm for partitioning problems with applications. In Hiroshi G. Okuno and Moonis Ali, editors, *New Trends in Applied Artificial Intelligence,*

20th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2007, volume 4570 of *Lecture Notes in Computer Science*, pages 217–226. Springer, 2007.

William J. Cook, William H. Cunningham, William R. Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization*. Wiley-Interscience, 1997.

Peter I. Cowling, Graham Kendall, and Eric Soubeiga. A hyperheuristic approach to scheduling a sales summit. In *PATAT '00 : Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III*, pages 176–190, London, UK, 2001. Springer-Verlag.

James P. Crutchfield. Is anything ever new ? considering emergence. In D. Pines G. Cowan and D. Melzner, editors, *Integrative Themes*, volume XIX of *Santa Fe Institute Studies in the Sciences of Complexity*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1994.

Joe Culberson. Holland's royal road and giga. GA-List v7n23, August 1993.

Charles Darwin. *The Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.* London : John Murray, 6th edition, 1872.

Mohammad Mehdi Dastani. *Languages of Perception*. PhD thesis, Institute for Logic, Language and Communication, Universiteit van Amsterdam, 1998.

Edwin D. de Jong, Dirk Thierens, and Richard A. Watson. Hierarchical genetic algorithms. In Xin Yao, Edmund Burke, Jose A. Lozano, Jim Smith, Juan J. Merelo-Guervós, John A. Bullinaria, Jonathan Rowe, Peter Tiňo Ata Kabán, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature - PPSN VIII*, volume 3242 of *LNCS*, pages 232–241, Birmingham, UK, 2004. Springer-Verlag.

Pierre De Lit, Emanuel Falkenauer, and Alain Delchambre. Grouping genetic algorithms : an efficient method to solve the cell formation problem. *Math. Comput. Simul.*, 51(3-4) :257–271, 2000.

Anne Defaweux. *Evolutionary Transitions as a Metaphor for Compositional Search : Definition and Evaluation of a New Optimisation Algorithm*. PhD thesis, Computational Modeling Lab, Computer Science Department, Vrije Universiteit Brussel, 2006.

Gianni Di Caro. *Ant Colony Optimization and its application to adaptive routing in telecommunication networks*. Phd thesis, IRIDIA, Faculé des Sciences Appliquées, Université Libre de Bruxelles, Brussels, Belgium, 2004.

Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. MIT Press (Bradford Books), 2004.

Russell C. Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43, 1995.

Emanuel Falkenauer. A new representation and operators for genetic algorithms applied to grouping problems. *Evolutionary Computation*, 2(2) : 123–144, 1994.

Emanuel Falkenauer. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2 :5–30, 1996.

Emanuel Falkenauer. *Genetic Algorithms and Grouping Problems*. John Wiley & Sons, Inc., New York, NY, USA, 1998.

Sir Ronald A. Fisher. *The Genetical Theory of Natural Selection*. Dover Publications Inc., 2nd - revised edition, 1958.

Stephanie Forrest and Melanie Mitchell. What makes a problem hard for a genetic algorithm ? some anomalous results and their explanation. *Machine Learning*, 13 :285–319, 1993.

Michael Randolph Garey and David S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. W. H. Freeman &amp ; Co., New York, NY, USA, 1979.

Matthew R. Glickman and Katia P. Sycara. Evolutionary search, stochastic policies with memory, and reinforcement learning with hidden state. In *ICML '01 : Proceedings of the Eighteenth International Conference on Machine Learning*, pages 194–201, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

Fred W. Glover and Gary A. Kochenberger. *Handbook of Metaheuristics (International Series in Operations Research & Management Science)*. Springer, January 2003.

David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1989.

David E. Goldberg, Kalyanmoy Deb, and Bradley Korb. Messy genetic algorithms : motivation, analysis, and first results. *Complex Systems*, 3(5) : 493–530, 1989.

Scott V. Gordon, Darrell L. Whitley, and Wim A. P. Böhm. Dataflow parallelism in genetic algorithms. In Reinhard Männer and Bernard Manderick, editors, *Parallel problem solving from nature 2*, pages 533–542, Amsterdam, 1992. North-Holland.

Richard L. Gregory. *Intelligent Eye*. London : Weidenfeld and Nicolson, 1970.

Paul Bryant Grosso. *Computer simulations of genetic adaptation : parallel subcomponent interaction in a multilocus model*. PhD thesis, Ann Arbor, MI, USA, 1985.

Georges R. Harik, Fernando G. Lobo, and David E. Goldberg. The compact genetic algorithm. In *IEEE Transactions on Evolutionary Computation*, volume 3, pages 523–528, 1999.

John H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, 1975.

John H. Holland. *Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press, April 1992.

John H. Holland. Royal road functions. GA-List v7n22, August 1993.

Holger H. Hoos and Thomas Stützle. *Stochastic Local Search : Foundations & Applications*. Elsevier, 2004.

Christian Igel and Marc Toussaint. On classes of functions for which no free lunch results hold. *Information Processing Letter*, 86(6) :317–321, 2003.

Terry Jones. A description of holland's royal road function. *Evol. Comput.*, 2(4) :409–415, 1994.

Terry Jones. *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, The University of New Mexico, Albuquerque, NM, 1995.

Leila Kallel, Bart Naudts, and Colin R. Reeves. *Theoretical aspects of evolutionary computing*, chapter Properties of fitness functions and search landscapes, pages 175–206. Natural Computing. Springer-Verlag, London, UK, 2001.

Scott Kirkpatrick, Charles Daniel Gelatt Jr., and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983.

Kurt Koffka. *Principles of Gestalt Psychology*. Harcourt (New York), 1935.

Pedro Larranaga. *Estimation of distribution algorithms : A new tool for evolutionary computation*. Kluwer Academic Publishers, 2002.

Tom Lenaerts. *Different Levels of Selection in Artificial Evolutionary Systems : Analysis and Simulation of Selection Dynamics*. PhD thesis, Department of Computer Science, Vrije Universiteit Brussel, 2003.

Worthy N. Martin, Jens Lienig, and James P. Cohoon. Island (migration) models : evolutionary algorithms based on punctuated equilibria. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, pages C6.3 :1–16. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

Keith E. Mathias and L. Darrell Whitley. Changing representations during search : A comparative study of delta coding. *Evol. Comput.*, 2(3) :249–278, 1994.

Michela Milano and Andrea Roli. Magma : A multiagent architecture for metaheuristics. *IEEE Trans. on Systems, Man and Cybernetics – Part B*, 34 (2) :925–941, April 2004.

Melanie Mitchell. *An introduction to genetic algorithms*. MIT Press, Cambridge, MA, USA, 1996.

Melanie Mitchell, Stephanie Forrest, and John H. Holland. The royal road for genetic algorithms : Fitness landscapes and ga performance. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life*, Cambridge, MA, 1992. MIT Press.

Nenad Mladenović and Pierre Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11) :1097–1100, 1997.

Alberto Moraglio. *Towards a Geometric Unification of Evolutionary Algorithms*. PhD thesis, Computer Science, University of Essex, UK, 2007.

Alberto Moraglio and Riccardo Poli. Abstract geometric crossover for the permutation representation. *IEEE Trans. Evolutionary Computation*, page submitted, 2005.

Alberto Moraglio, Yong-Hyuk Kim, Yourim Yoon, and Byung-Ro Moon. Geometric crossovers for multiway graph partitioning. *Evol. Comput.*, 15 (4) :445–474, 2007.

David E. Moriarty, Alan C. Schultz, and John J. Grefenstette. Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research*, 11 :241–276, 1999.

Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization : Algorithms and Complexity*. Dover Publications, 1998.

Martin Pelikan. *Hierarchical Bayesian Optimization Algorithm : Toward a New Generation of Evolutionary Algorithms*. Springer, 2005.

Martin Pelikan, David E. Goldberg, and Fernando G. Lobo. A survey of optimization by building and using probabilistic models. *Comput. Optim. Appl.*, 21(1) :5–20, 2002.

Christophe Philemotte. "good" observers enhance sga exploration. In Mario Giacobini and Jano van Hemert, editors, *First European Graduate Student Workshop on Evolutionary Computation*, pages 59–72, Budapest, Hungary, 2006. URL http://evonet.lri.fr/eurogp2006/EvoPhD2006.pdf.

Christophe Philemotte. A stochastic routing policy perspective for an operator-level hyperheuristic. Technical Report TR/IRIDIA/2008-018, IRIDIA, CoDE, Université Libre de Bruxelles, August 2008.

Christophe Philemotte and Hugues Bersini. Coevolution of effective observers and observed multi-agents system. In *Advances in Artificial Life, 8th European Conference, ECAL 2005, Canterbury, UK, September 5-9, 2005, Proceedings*, volume 3630 of *Lecture Notes in Computer Science*, pages 785–794. Springer, 2005a.

Christophe Philemotte and Hugues Bersini. Intrinsic emergence boosts adaptive capacity. In *GECCO '05 : Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 559–560, New York, NY, USA, 2005b. ACM Press.

Christophe Philemotte and Hugues Bersini. How an optimal observer can smooth a landscape. In *IEEE Congress on Evolutionary Computation 2006*, pages 2293–2300. IEEE, 2006a.

Christophe Philemotte and Hugues Bersini. How an optimal observer can collapse the search space. In *GECCO '06 : Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1273–1280, New York, NY, USA, 2006b. ACM Press.

Christophe Philemotte and Hugues Bersini. A gestalt genetic algorithm : less details for better search. In Hod Lipson, editor, *GECCO '07 : Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1328–1334, New York, NY, USA, 2007. ACM.

Christophe Philemotte and Hugues Bersini. The gestalt heuristic : learning the right level of abstraction to better search the optima. Technical Report TR/IRIDIA/2008-021, IRIDIA, CoDE, Université Libre de Bruxelles., October 2008. submitted to Artificial Intelligence Journal.

Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A field guide to genetic programming*. Published via `http ://lulu.com` and freely available at `http ://www.gp-field-guide.org.uk`, 2008. URL http://www.gp-field-guide.org.uk. (With contributions by J. R. Koza).

Nicholas J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5 :183 – 205, 1991.

Nicholas J. Radcliffe and Patrick D. Surry. Fitness variance of formae and performance prediction. In L. Darrell Whitley and Michael D. Vose, editors, *Proceedings of the Third Workshop on Foundations of Genetic Algorithms*, pages 51–72. Morgan Kaufmann, 1994.

Soraya B. Rana and L. Darrell Whitley. Bit representations with a twist. In Thomas Bäck, editor, *Proceedings of the 7th International Conference on Genetic Algorithms, ICGA 1997*, pages 188–195. Morgan Kaufmann, 1997.

Colin R. Reeves. Landscapes, operators and heuristic search. *Annals of Operations Research*, 86 :473 – 490, 1999.

Christian M. Reidys and Peter F. Stadler. Combinatorial landscapes. *SIAM Rev.*, 44(1) :3–54, 2002.

Peter Ross. Hyper-heuristics, 2005.

Gian-Carlo Rota. The number of partitions of a set. *American Mathematical Monthly*, 71(5) :498—504, 1964.

Franz Rothlauf. *Representations for Genetic and Evolutionary Algorithms*. Springer, 2nd edition, 2006.

Jonathan Rowe, Darrell Whitley, Laura Barbulescu, and Jean-Paul Watson. Properties of gray and binary representations. *Evolutionary Computation*, 12(1) :47–76, 2004.

Reuven Y. Rubinstein. Cross-entropy and rare events for maximal cut and partition problems. *ACM Trans. Model. Comput. Simul.*, 12(1) :27–53, 2002.

Conor Ryan, J. J. Collins, and Michael O'Neill. Grammatical evolution : Evolving programs for an arbitrary language. In Wolfgang Banzhaf, Riccardo Poli, Marc Schoenauer, and Terence C. Fogarty, editors, *Genetic Programming, First European Workshop, EuroGP'98, Paris, France, April 14-15, 1998, Proceedings*, volume 1391 of *Lecture Notes in Computer Science*, pages 83–96. Springer, 1998.

J. David Schaffer and Larry J. Eshelman. On crossover as an evolutionary viable strategy. In R.K. Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 61–68. Morgan Kaufmann, 1991.

Brian J. Scholl. Objects and attention : the state of the art. *Cognition*, 80(1-2) : 1–46, June 2001.

Cosma Rohilla Shalizi. *Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata*. PhD thesis, Physics Department, University of Wisconsin at Madison, 2001.

Zbigniew Skolicki and Kenneth A. De Jong. Improving evolutionary algorithms with multi-representation island models. In *Parallel Problem Solving from Nature - PPSN VIII*, volume 3242 of *Lecture Notes in Computer Science*, pages 420–429. Springer, 2004.

Zbigniew Maciej Skolicki. *An Analysis of Island Models in Evolutionary Computation*. PhD thesis, Department of Computer Science, George Mason University, 2007.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning : An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, 1998.

Allan Tucker, Jason Crampton, and Stephen Swift. Rgfga : An efficient representation and crossover for grouping genetic algorithms. *Evol. Comput.*, 13(4) :477–499, 2005.

Jano I. van Hemert. Evolving combinatorial problem instances that are difficult to solve. *Evolutionary Computation*, 14(4) :433–462, 2006.

Michael D. Vose. *The Simple Genetic Algorithm : Foundations and Theory*. MIT Press, Cambridge, MA, USA, 1998.

Chris Walshaw. Multilevel refinement for combinatorial optimisation problems. *Annals of Operations Research*, 131 :325–372, 2004.

Jean-Paul Watson, Laura Barbulescu, L. Darrell Whitley, and Adele E. Howe. Contrasting structured and random permutation flow-shop scheduling problems : Search space topology and algorithm performance. *INFORMS Journal on Computing*, 14(2) :98–123, 2002.

Richard A. Watson. Analysis of recombinative algorithms on a non-separable building-block problem. In Worthy N. Martin William M. Spears, editor, *Proceedings of the Sixth Workshop on Foundations of Genetic Algorithms*, pages 69–90. Morgan Kaufmann, 2001.

Richard A. Watson. *Compositional evolution : interdisciplinary investigations in evolvability, modularity, and symbiosis*. PhD thesis, Brandeis University, 2002. Adviser-Jordan B. Pollack.

Richard A. Watson. *Compositional Evolution : The Impact of Sex, Symbiosis, and Modularity on the Gradualist Framework of Evolution*. Vienna Series in Theoretical Biology. The MIT Press, 2006.

Richard A. Watson, Greg Hornby, and Jordan B. Pollack. Modeling building-block interdependency. In *PPSN V : Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pages 97–108, London, UK, 1998. Springer-Verlag.

Max Wertheimer. *A Source Book of Gestalt Psychology*, chapter Laws of Organization in Perceptual Forms, pages 71–88. New York : The Humanities Press, 1938. English translation of the original paper published as Untersuchungen zur Lehre von der Gestalt II (1923).

Max Wertheimer. *Productive thinking*. Harper (New York), 1959.

Darrell Whitley, T. Starkweather, and D. Fuquay. Scheduling problems and traveling salesman : The genetic edge recombination operator. In *Proceedings of the 3rd International Conference on Genetic Algorithms (ICGA)*, pages 133–140, 1989.

R. Paul Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, Department of Computer Science, George Mason University, 2003.

David Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Trans. Evolutionary Computation*, 1(1) :67–82, 1997.

Sewall Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In D. F. Jones, editor, *Proceedings of the Sixth International Congress on Genetics*, volume 1, pages 356–366, 1932.

Song-Chun Zhu. Embedding gestalt laws in markov random fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(11) :1170–1187, 1999.

Mark Zlochin, Mauro Birattari, Nicolas Meuleau, and Marco Dorigo. Model-based search for combinatorial optimization : A critical survey. *Annals of Operations Research*, 131(1–4) :373–395, 2004.